n° d'ordre :

année : 1980

THESE

présentée devant l'Université Claude Bernard

pour obtenir le diplome de docteur ingénieur

Spécialité : Informatique et automatique appliquées aux systèmes industriels et de gestion

par J.-Lucien RASCLE

CAO : UNE AIDE GRAPHIQUE A LA MODELISATION DES SYSTEMES DYNAMIQUES

soutenue le 28 octobre 1980 devant la commission d'examen

MM.	A. DUSSAUCHOY	Président
	P. COUEIGNOUX	Examinateurs
	J. FAVREL	
	A. MATHON	
	M. SCHNEIDER	

n° d'ordre :

année : 1980

THESE

présentée devant l'Université Claude Bernard

pour obtenir le diplome de docteur ingénieur

Spécialité : Informatique et automatique appliquées aux systèmes industriels et de gestion

par J.-Lucien RASCLE

CAO : UNE AIDE GRAPHIQUE A LA MODELISATION DES SYSTEMES DYNAMIQUES

soutenue le 28 octobre 1980 devant la commission d'examen

MM.	A. DUSSAUCHOY	Président
	P. COUEIGNOUX	Examinateurs
	J. FAVREL	
	A. MATHON	
	M. SCHNEIDER	

UNIVERSITE CLAUDE BERNARD - LYON I -

Président honoraire : Mr le Pr J. BOIDIN

Président : M. le Pr D. GERMAIN
Premier Vice-Président : M. le Pr M. DUFAY
Deuxième Vice-Président : M. J.C. DUPLAN, M.A.
Troisième Vice-Président : Mle ECHALLON, étudiante
Secrétaire Général de l'Université : M. J. RAMBAUD, Administrateur Civil

UNITES D'ENSEIGNEMENT ET DE RECHERCHE (U.E.R.)

U.E.R. de Médecine GRANGE-BLANCHE ..... : Monsieur le Pr Paul ZECH

U.E.R. de Médecine ALEXIS-CARREL ..... : Monsieur le Pr René MORNEX

U.E.R. de Médecine LYON-NORD ..... : Monsieur Yves MINAIRE, M.C.A.

U.E.R. de Médecine LYON-SUD ..... : Monsieur le Pr Jean NORMAND

U.E.R. Faculté de Pharmacie ..... : Monsieur le Pr C.A. BIZOLLON

U.E.R. de Techniques de Réadaptation ..... : Monsieur le Pr Alain MORGON

U.E.R. de Biologie Humaine ..... : Mr Jean-Pierre REVILLARD, M.C.A.

U.E.R. E.P.S. ..... : Mr Albert MILLON, Pr d'E.P.S.

U.E.R. Faculté d'Odontologie de Lyon ..... : Monsieur le Pr Jean PARRET

U.E.R. de Mathématiques ..... : Monsieur le Pr Philippe PICARD

U.E.R. de Physique ..... : Monsieur le Pr Jean DELMAU

U.E.R. de Chimie et Biochimie ..... : Madame Annick VARAGNAT, M.A.

U.E.R. des Sciences de la Nature ..... : Monsieur le Pr Yves LEMOIGNE

U.E.R. de Sciences Physiologiques ..... : Mle le Pr J.F. WORBE

U.E.R. de Physique Nucléaire ..... : Monsieur le Pr Mark GUSAKOW

I.U.T. I ..... : Monsieur le Pr Albert VILLE

I.U.T. II ..... : Mr J. GALLET, Directeur ENSAM

Observatoire de Lyon ..... : Mr G. MONNET, Astronome Adjoint

U.E.R. de Mécanique ..... : Mle le Pr G. COMTE-BELLOT

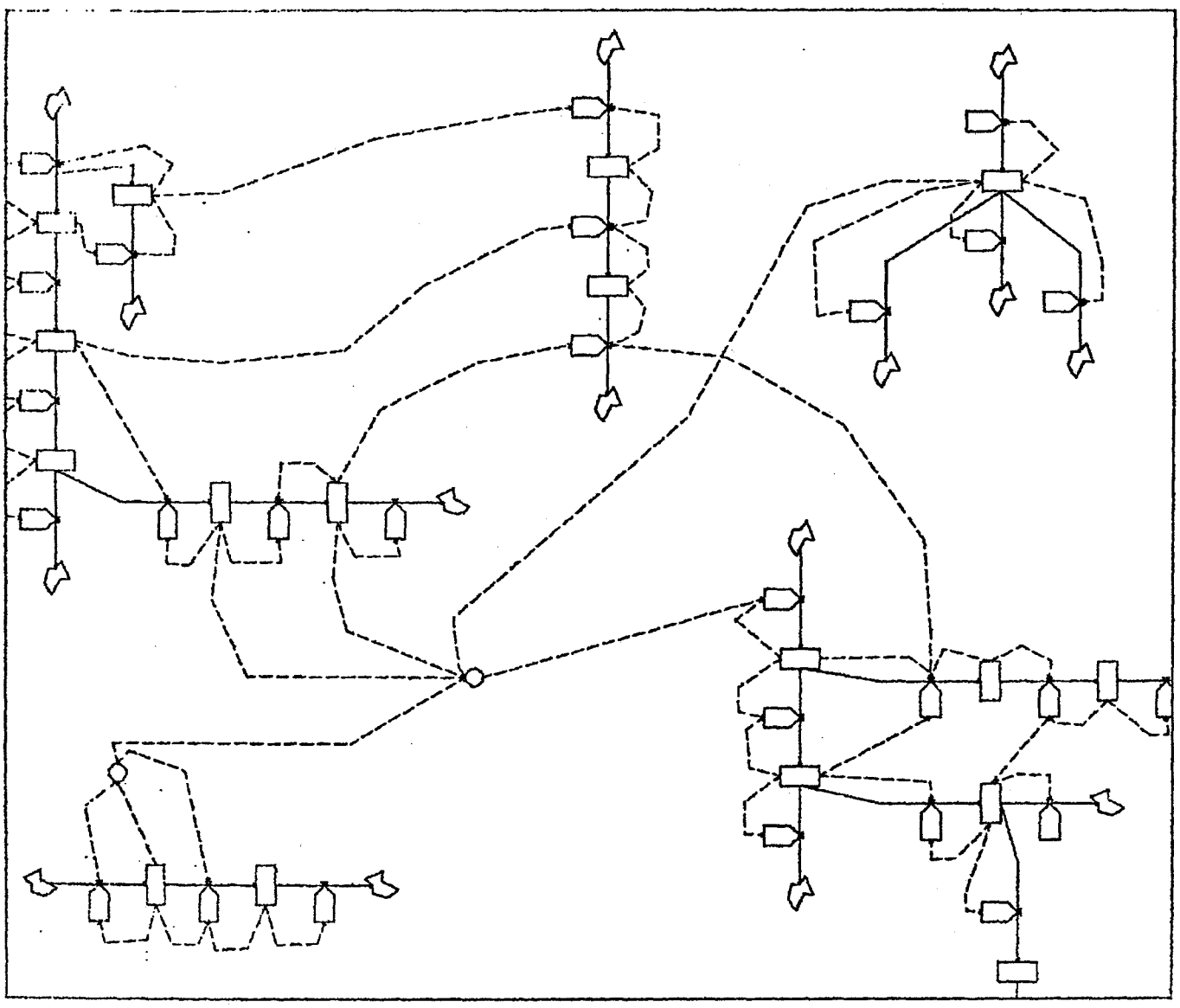
Je remercie ici,

Alain DUSSAUCHOY, professeur d'informatique à l'Université Claude Bernard, Lyon I, qui a dirigé ce travail ;
Albert MATHON, maître de recherche, Directeur du Département Entreprise et Travaux à l'Ecole des Mines de Saint-Etienne dans l'équipe de qui j'ai travaillé ; Ph. COUEIGNOUX, maître de recherche à l'Ecole des Mines de Saint-Etienne pour les nombreux conseils qu'il m'a prodigués ; J. FAVREL Professeur à l'INSA de Lyon et M. SCHNEIDER Professeur au CUST de Clermont Ferrand qui ont accepté de faire partie du jury ; André LOUBET ; Fleury VELAY, et Louis DARLES qui ont imprimé ce texte.

et particulièrement Jean-Pierre JOURDAN, Ingénieur Civil des Mines, qui a mené des travaux similaires avant ceux-ci et qui m'a soutenu tout au long de ce travail.

DYNAMINE GRAPHIQUE

FIN	MODIF	PUITS	CSTE	ZOOM+
	PLACE	VANNE	LIAIS	ZOOM-
	HOR	NIVEAU	TEXTE	FENETR
	VER	AUXIL		



Une aide graphique à la modélisation des systèmes dynamiques

SOMMAIRE.

IV Conclusion et remarques diverses

Annexes:

Exemple d'utilisation

A Diagramme causal

B diagramme des flux

C Ecriture des équations

D Exécution

E Résultats

L'environnement matériel

I. Introduction.

Le travail présenté ici consiste en l'édification d'un système interactif et graphique de modélisation et de simulation. Ce travail vient après bien d'autres dont les plus approchants sont DYNASIS de JARSCH et surtout GRIPPS de HUDITZ. L'originalité tient en une plus grande automatisation (une décharge du travail sur la machine plus importante) du dessin et du traitement.

Les raisons qui ont fondé ce système sont des besoins différents. D'abord un besoin pédagogique, la simulation à l'aide de la dynamique des systèmes est une bonne approche des problèmes des systèmes complexes et des comportements contre intuitifs. Néanmoins, si la dynamique des systèmes est pédagogique, l'ennui est qu'avec les compilateurs habituels, le fond du problème est éludé par l'aspect technique rébarbatif de la simulation: nombreux passages pour la correction des erreurs de compilation entre autres. D'autre part, il y avait une totale déconnection entre les travaux préliminaires, détermination des limites du modèle, dessin du graphe etc..., et la simulation proprement dite. Il y avait donc là un hiatus à combler. Ces remarques nous ont donc conduits à créer un système où sont intégrés et reliés les aspects de modélisation et de simulation tout en déchargeant sur la machine le plus de tâches techniques annexes possible.

Un autre besoin est la demande de systèmes permettant de rendre la programmation plus rapide, plus fiable et accessoirement (mais n'est ce pas là le plus important ?) plus agréable. A l'heure où on se pose la question de savoir si le manque d'informaticiens ne va pas pénaliser notre développement informatique et donc économique, il est utile de rechercher des solutions qui permettent de programmer plus vite et plus sainement, ce qui est une branche de l'alternative: former plus d'informaticiens (ce qui risque d'in-

troduire un comportement oscillant de l'emploi entre des pointes et des creux) ou créer des outils remplaçant les spécialistes manquants. Parmi ces outils, la C.A.O. a une place privilégiée, mais jusqu'à présent elle n'a été conçue que pour des domaines bien particuliers:

-étude de structures et de leurs déformations (éléments finis)

-architecture

-électronique et micro-électronique

etc...

mais n'a été que rarement appliquée à la programmation en générale. Pourtant dans ce domaine, le problème crucial est de passer d'une pratique empirique à une pratique plus scientifique (passer de la sorcellerie à l'ingénierie). Des essais ont été effectués très tôt dans les années 60 mais n'ont eu aucune retombée du fait du prix des terminaux et du nombre restreint d'applications interactives. Mais actuellement, le prix des matériels (en baisse constante), le prix de la main d'oeuvre (en hausse constante) et le manque de spécialistes font que de telles solutions seront remises au goût du jour. Ces solutions sont:

-emploi du graphique pour la programmation: liaison directe entre le schéma de la structure du programme et le programme lui-même.

-aspect interactif et spécialisé: éditeur spécialisé permettant la détection immédiate des fautes de syntaxe entre autres.

De telles procédures n'élimineront pas toutes les causes d'erreur mais permettront certainement d'abaisser le taux d'erreur de façon cruciale. Ce fait a été confirmé par l'emploi de dynamine graphique

qui a montré que le temps de mise au point d'un modèle est abaissé de manière importante (facteur 2 ou 3) entre une méthode batch et la méthode interactive graphique. Il est évident que ce système ne permet qu'une extrapolation pour la programmation générale mais il donne des indications intéressantes sur les gains de temps et les méthodes à employer.

II La Dynamique des Systèmes : Le langage

La dynamique des systèmes est directement issue des travaux de cybernétique et d'automatique des années 1950 et utilise les notions introduites par ces deux matières. Le principe de base qui sous-tend cette discipline correspond à la Gestalt Theorie et indique que:

"le tout est supérieur à la somme des parties"

il faut en effet tenir compte des interactions nées de la juxtaposition des parties du système. C'est ce qu'on appelle couramment la synergie.

Pour la dynamique des systèmes, les systèmes sont des ensembles de composants liés entre eux par des relations de cause à effet en vue d'une finalité. La particularité de tels systèmes est qu'en général le graphe de liaisons les décrivant n'est pas un arbre mais un graphe quelconque (le plus souvent connexe) comportant des boucles: boucles de rétro-action ou feed-back. La seule hiérarchie existante en principe est celle de la puissance de l'influence d'un composant sur un autre.

A Diagramme causal

La première étape dans la modélisation sera d'établir le diagramme causal, c'est à dire le graphe des liaisons (des influences) des composants les uns avec les autres.

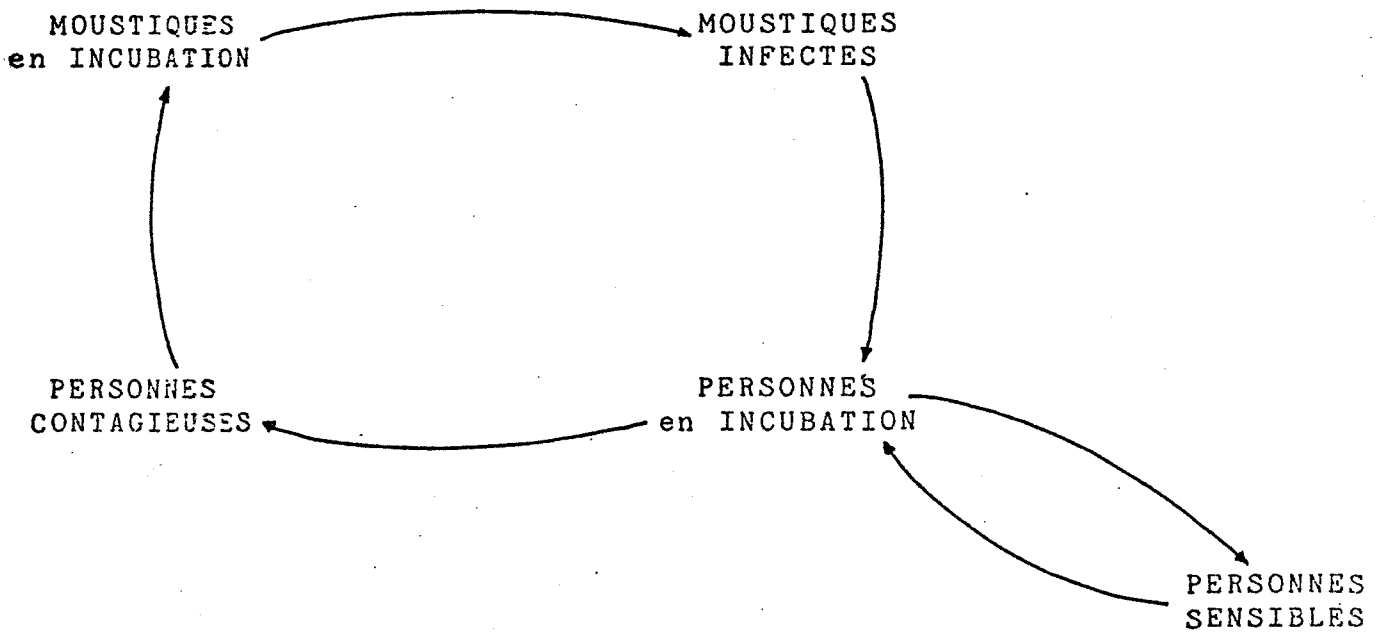


Diagramme causal du modèle d'une épidémie de fièvre jaune

(diagramme simplifié)

Le schéma précédent signifie en "clair":

" le nombre de personnes contagieuses influe directement sur le nombre de moustiques en incubation qui influe sur le nombre de moustiques infectés, qui influe sur le nombre de personnes en incubation qui influe sur le nombre de personnes contagieuses. D'autre part le nombre de personnes en incubation influe sur le nombre de personnes sensibles qui influe en retour sur le nombre de personnes en incubation."

La tournure de cette phrase explique pourquoi "clair" est mis entre guillemets et pourquoi les diagrammes causaux sont très utilisés.

Pour définir un peu plus finement le modèle dans sa structure, on peut affecter à chaque relation le signe correspondant au type d'influence:

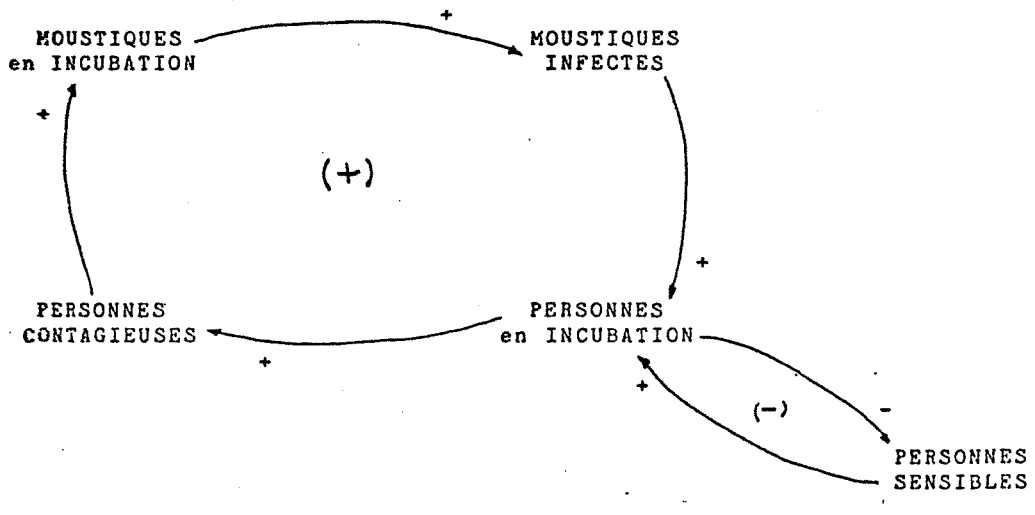
- signe + si la variable influencée B varie dans le même sens que la variable A qui est la cause de cette variation.

  si A augmente, B augmente

- signe - si la variable influencée B varie en sens inverse de la variable A qui l'influence.

  si A augmente, B diminue

D'où le nouveau schéma:



Grâce à ce schéma on peut déjà avoir une idée du comportement qualitatif du système. En effet, on voit clairement qu'il existe deux boucles

une boucle faisant intervenir les moustiques

une boucle où n'interviennent que les humains

Si on analyse séparément ces deux boucles, on voit que la première ne comporte que des liaisons affectées d'un signe +, ce qui signifie que si une variable est modifiée dans un sens, toutes les autres variables de la boucle varieront dans le même sens pour influencer en retour la première et donc accentuer sa variation. Par exemple une augmentation du nombre de moustiques infectés induira une augmentation des moustiques en incubation, par le biais des personnes en incubation et des personnes contagieuses, ce qui fera augmenter le nombre de moustiques infectés: c'est une boucle qui décrit un comportement "explosif". Si cette seule boucle décrivait le système nous ne pourrions avoir que 3 types de comportements:

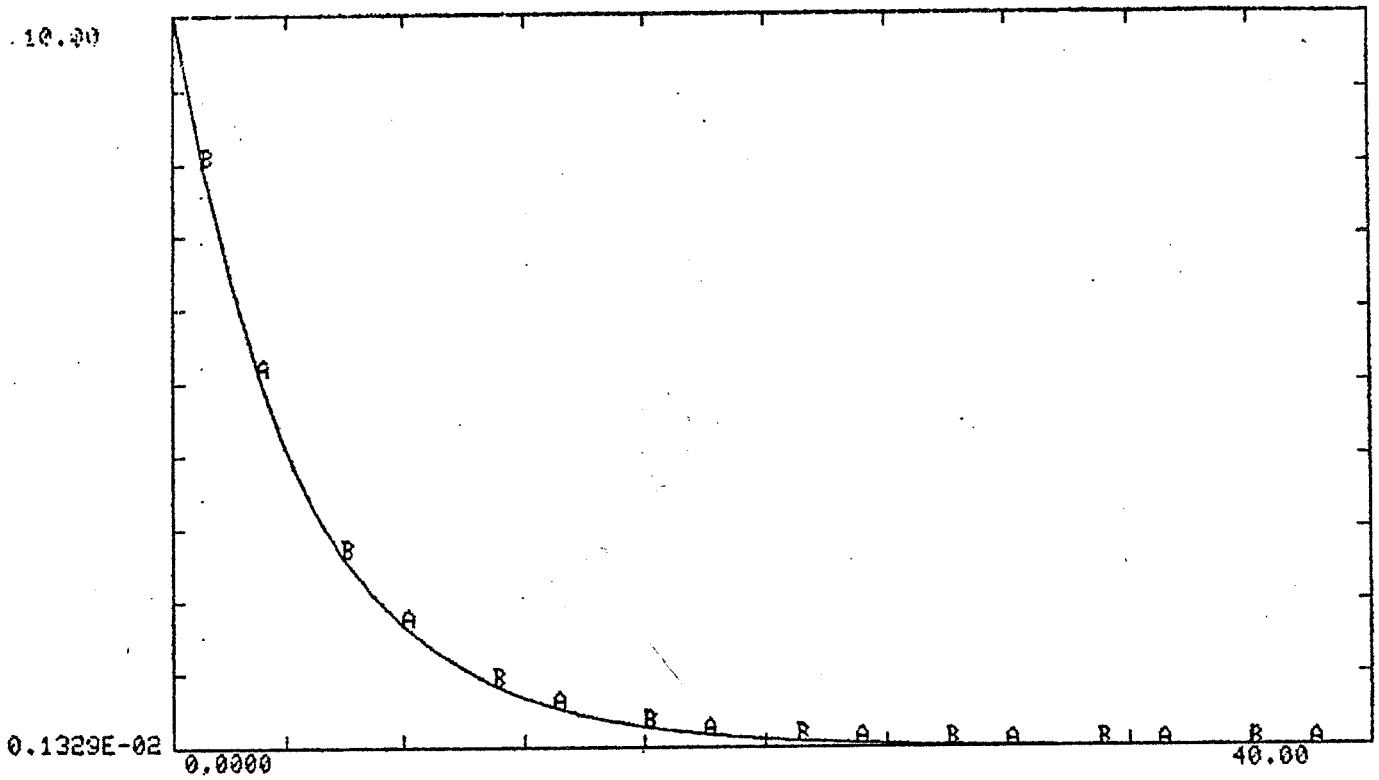
- équilibre instable (ce qui signifie que dans les conditions étudiées le système est en équilibre mais que la moindre perturbation de ces conditions le ferait retomber dans un des deux comportements suivants)
- "explosion" (croissance exponentielle)
- décroissance exponentielle

(voir fig. 1)

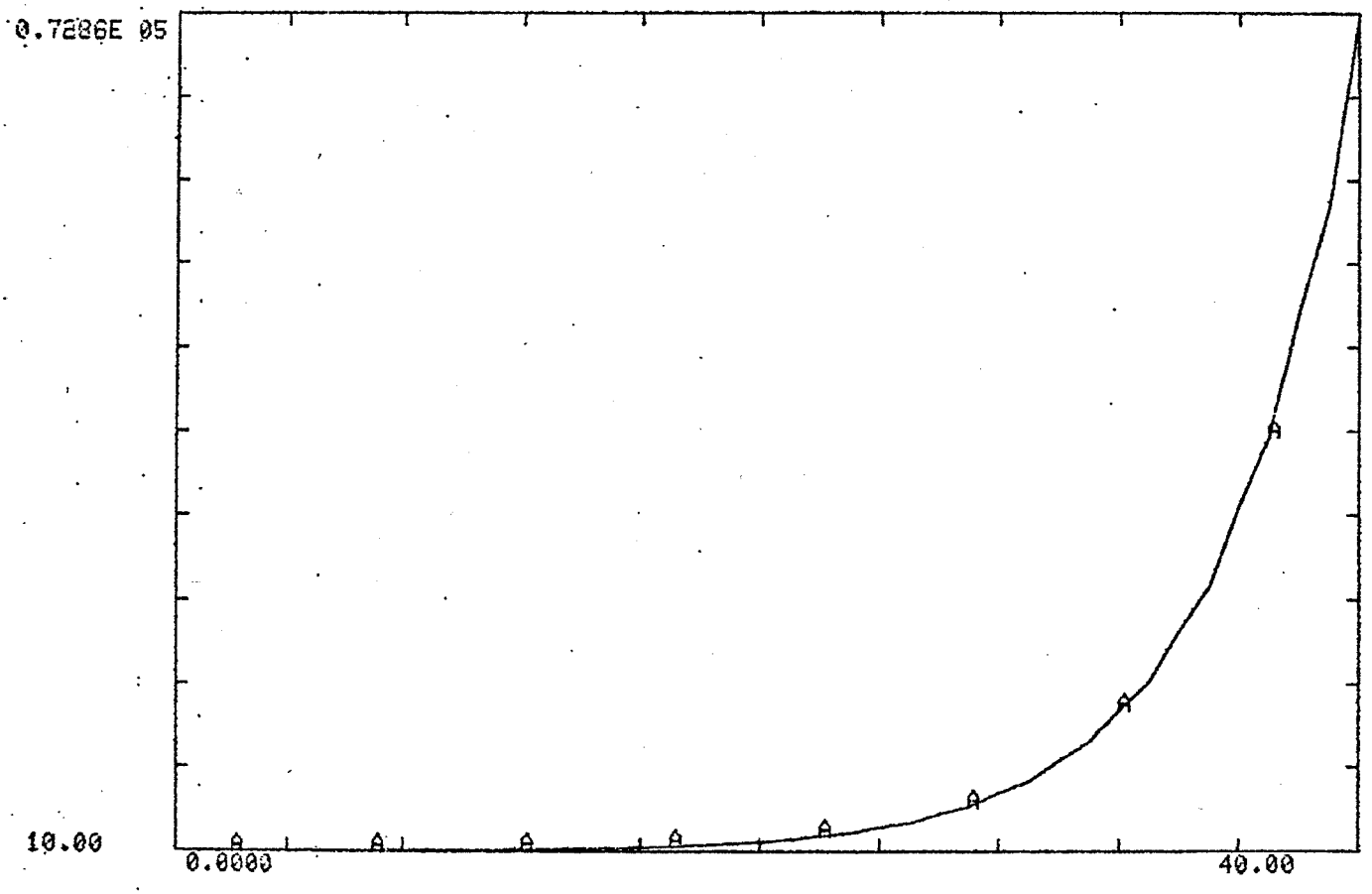
Ce genre de boucle est appelé boucle positive ou feed-back positif. Il sert à décrire des effets "boules de neige".

La deuxième boucle du modèle décrit un comportement tout à fait différent. La suite des liaisons fait apparaitre un nombre impair de liaisons affectées d'un signe -. La variation possible d'un élément est contrariée en retour par la variation des autres éléments:

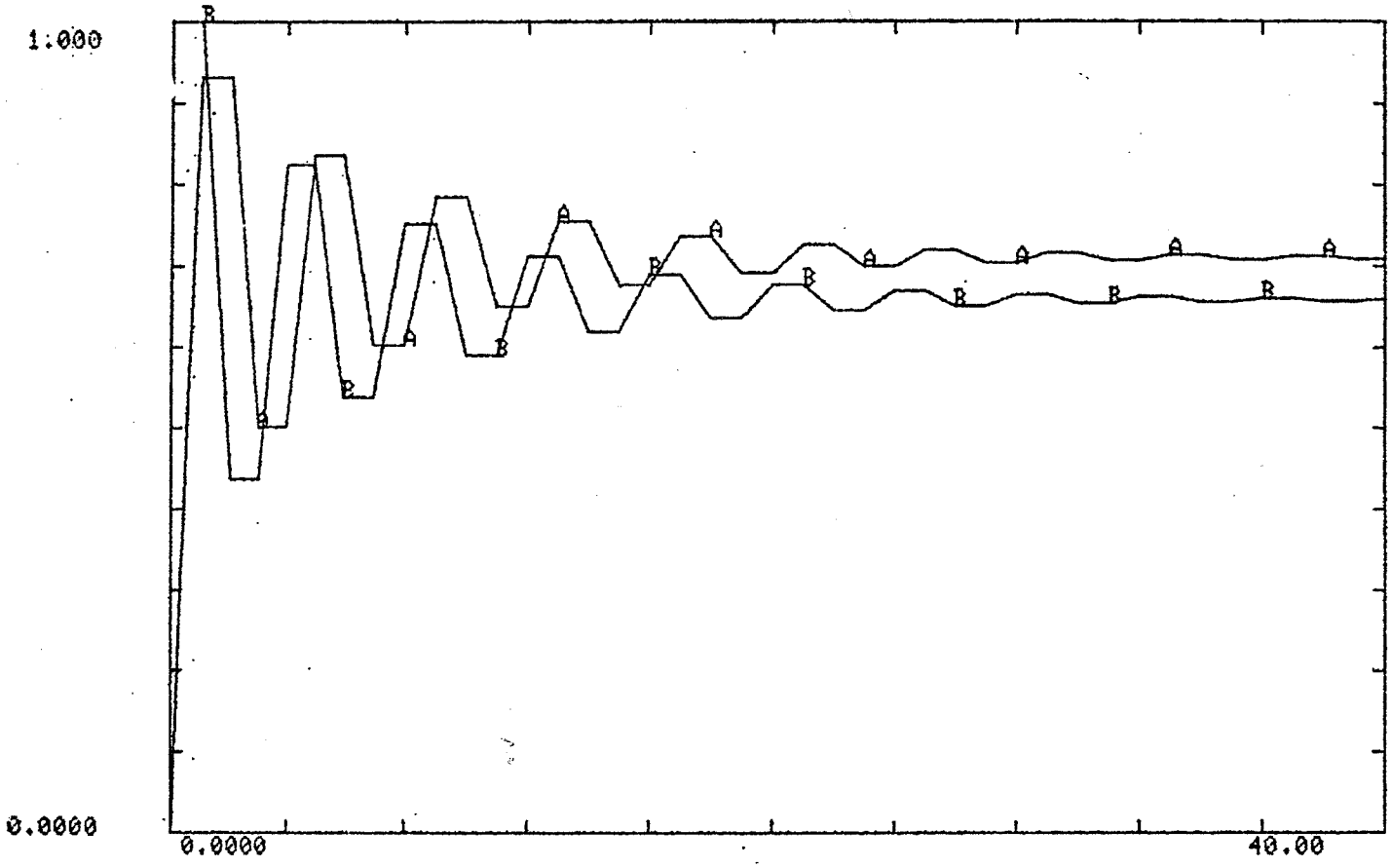
si le nombre de personnes en incubation augmente, le nombre de personnes sensibles diminue, ce qui fait qu'en retour le



A A MIN 0.1329E-02 MAX 10.00
B C MIN 0.1329E-02 MAX 10.00



A A MIN 10.00 MAX 0.7286E 05



A A MIN 0.0000 MAX 0.9320
B B MIN 0.4371 MAX 1.000

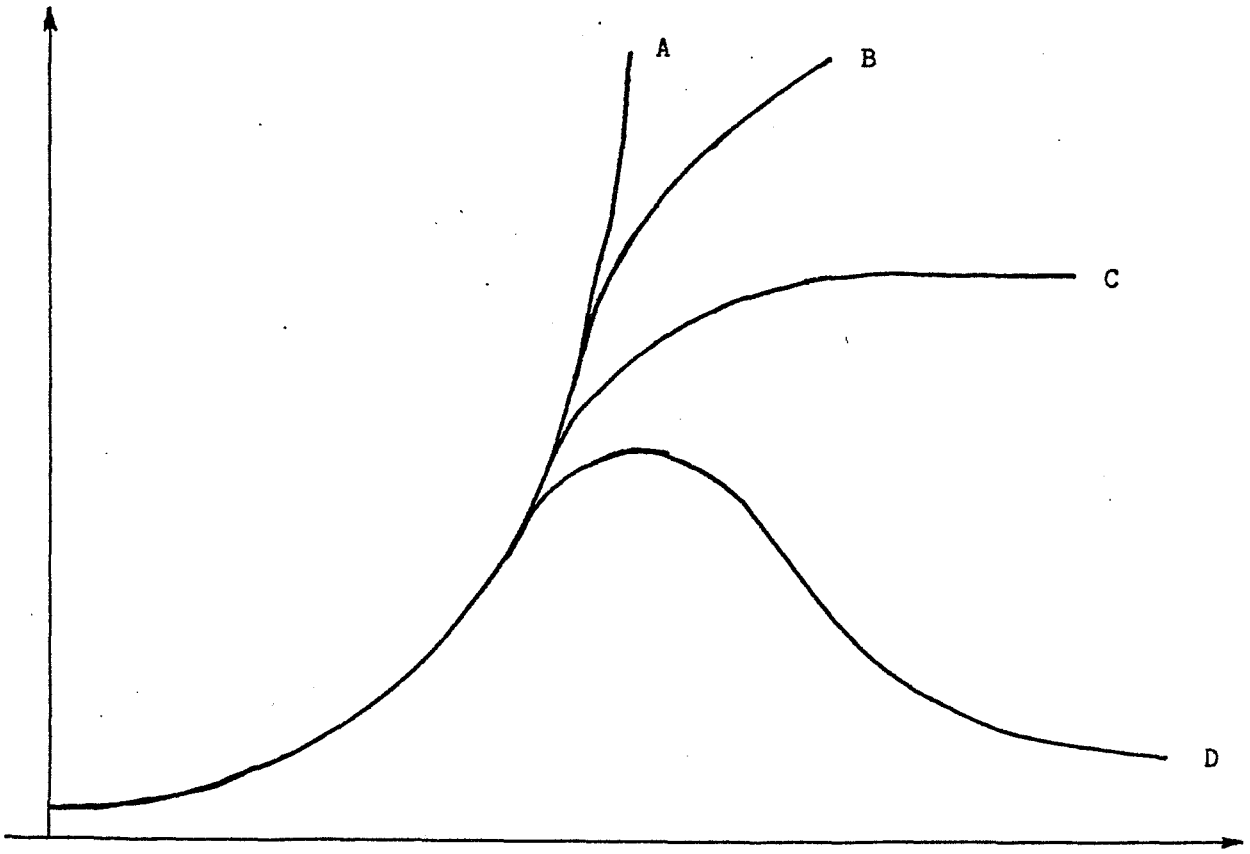
Fig. 2

nombre de personnes en incubation ne pourra que diminuer.

Ce type de boucle, appelé boucle négative ou feed-back négatif, caractérise les structures de contrôle ou de régulation des systèmes. En effet, ces boucles font que le système ne peut qu'arriver à une situation d'équilibre. On a le comportement suivant:

oscillation amorties vers un équilibre (fig.2)

L'étude de la combinaison des deux types de boucles ne peut pas donner grand chose sans renseignements supplémentaires. Tout ce qu'on peut dire c'est que la présence d'une boucle négative est encourageante car une position d'équilibre peut être trouvée. Les différentes trajectoires pouvant s'apparenter aux suivantes:



A croissance exponentielle
B courbe en S, croissance lente
C croissance puis stagnation
D croissance puis déclin

On possède donc à partir de la seule représentation qualitative du système (diagramme causal signé) d'une typologie de comportements. Mais cela est insuffisant pour pouvoir décrire le comportement particulier d'un système complexe, il est nécessaire pour cela de quantifier les relations. En effet une même structure peut conduire à un comportement A ou B, il importe donc de parvenir à élucider les conditions (paramètres ou relations) qui font passer le "même" système de A en B.

B Les différents types de variables

A ce stade, il est nécessaire de faire une étude plus quantitative du système. Pour cela la dynamique des systèmes supposera qu'il existe plusieurs types de variables décrivant les composants du modèle:

- les variables d'état du système (les niveaux)
- les flux
- les variables de décision ou de contrôle

Les variables d'état décrivent à chaque instant l'état du modèle et intègrent donc les résultats des actions entreprises. Elles correspondent à des composants qui intégrent des flux (d'où leur nom de niveau), ces niveaux sont donc alimentés par les flux qui sont des valeurs instantanées. Ces flux sont fixés par les variables de contrôle et de décision: la dynamique des systèmes, c'est de la plomberie... Grâce à ces éléments simples on peut déjà décrire des systèmes complexes avec une grande souplesse d'emploi.

Pour l'instant nous avons tenu le temps à l'écart de nos préoccupations, ce qui signifie entre autre que nous supposons que les transferts d'un élément à un autre se font de manière quasi instantanée, du moins vis à vis de la perception du temps qu'a le modèle, c'est à dire du plus petit intervalle de temps pris en compte par le modèle. C'est une hypothèse simpliste et contraignante, aussi est-il nécessaire d'introduire un nouveau facteur: les délais qui rendent compte du temps de transmission des matières ou des informations.

La structure de nos modèles se complique donc et permet d'introduire des éléments nouveaux qui n'apparaissent pas sur le diagramme causal: le temps et les délais qui font que les modèles qui nous occupent sont dynamiques.(nous ne parlons pas ici de leur structure qui reste figée, mais de leur comportement).

C Les équations.

Avant d'aborder l'écriture des équations, il est nécessaire de savoir de quelle façon s'effectuera la simulation. Pour simuler le comportement du système dans le temps, nous découperons le temps en intervalles dt constants suffisamment petits pour que durant lesquels les différentes variables autres que les variables d'état resteront constantes, la mise à jour se faisant tous les dt en fonction des valeurs précédentes.

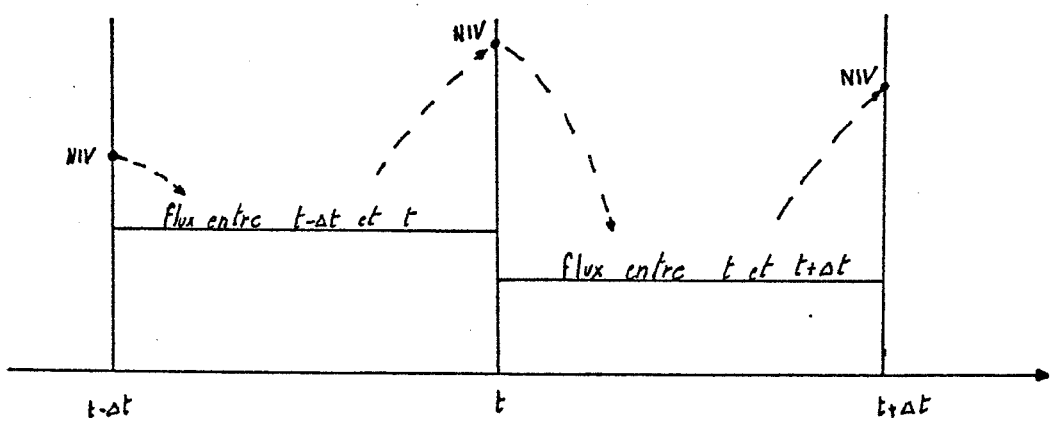
Sachant ceci les équations de niveaux (variables d'état) s'écriront très naturellement:

NIV(t+dt) = NIV(t)+dt*(Σ (flux en entrée entre t et t+dt) -Σ(flux en sortie entre t et t+dt))

Les équations des autres variables dépendent du système étudié et du modèle qu'on en fait. Elles seront du type:

VAR(t+dt) = f(variables du système à t et entre t et t+dt)

Nous avons donc la situation suivante pour les relations temporelles entre les variables:



Pour simplifier l'écriture des équations nous utiliserons les notations suivantes:

NIV(J) et NIV(K) pour les niveaux et les variables auxiliaires

J désignant l'instant précédent soit dans les notations précédentes t.(où tout est connu)

K désignantl'instant présent soit t+dt.(où l'on cherche à déterminer la valeurde l'ensemble des variables d'état)

FL(J) et FL(K) pour les flux.

J désignant dans ce cas l'intervalle de temps t-dt,t

K désignant l'intervalle t,t+dt

Les équations s'écriront alors:

NIV(K) = NIV(J)+DT*(Σ flux) pour les niveaux

FL(K) = f(variables du système en J et K) pour les autres types de variables.

Il faut remarquer que certaines écritures ne sont pas permises pour des raisons tant logiques que techniques. En effet le groupe d'équations:

AUX1(K) = f(AUX2(K),...)

AUX2(K) = g(AUX1(K),...)

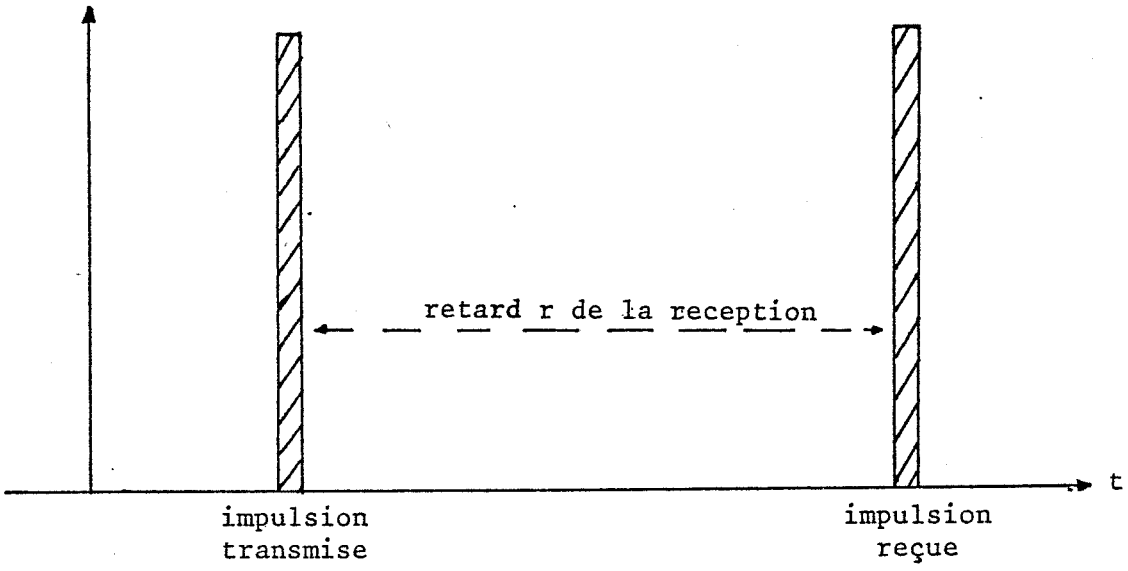
n'est pas évaluable et de toute façon ne signifie pas grand chose au niveau physique car il suppose que les interactions sont instantanées.

D. Les délais.

La transmission entre deux composants des modèles n'étant pas instantanée, il est nécessaire d'avoir recours à des délais. Ceux-ci seront de deux sortes:

- retard pur: la quantité transmise à l'instant t arrive en un seul bloc après un temps r qui est le retard. Ce retard peut évidemment varier suivant l'instant t d'envoi ou suivant n'importe quel(s) autre(s) paramètre(s).

- délai exponentiel: une quantité transmise à l'instant t arrive en "ordre dispersé" après un retard r donné. Dans ce cas le retard r ne peut varier (pour des raisons essentiellement techniques). Dans ce type de délai ce qui est aléatoire est l'ordre et l'instant d'arrivée de chaque "morceau" élémentaire de la quantité initialement transmise. Ce que nous percevons au niveau du modèle c'est en fait la structure macroscopique déterministe composée des faits microscopiques aléatoires que sont les arrivées des "morceaux".



Retard pur

Fig. 3

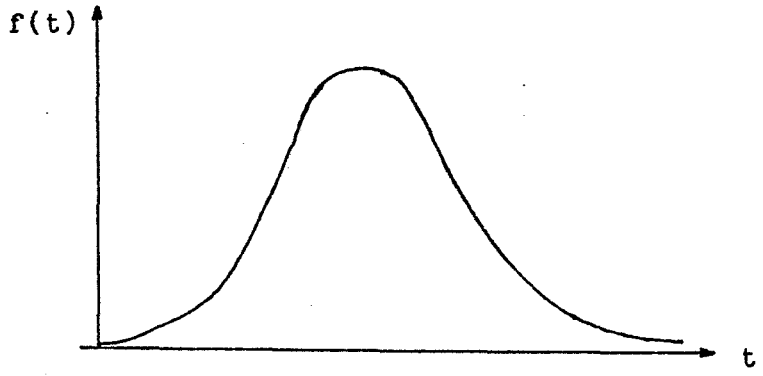
Les retards purs ne présentent aucune difficulté de compréhension: l'arrivée de la quantité transmise subit un décalage dans le temps par rapport à l'instant d'émission. Les délais exponentiels sont des phénomènes plus délicats à appréhender. Deux points de vue peuvent en expliquer le mécanisme:

- un point de vue automatisme qui en fait des filtres linéaires.

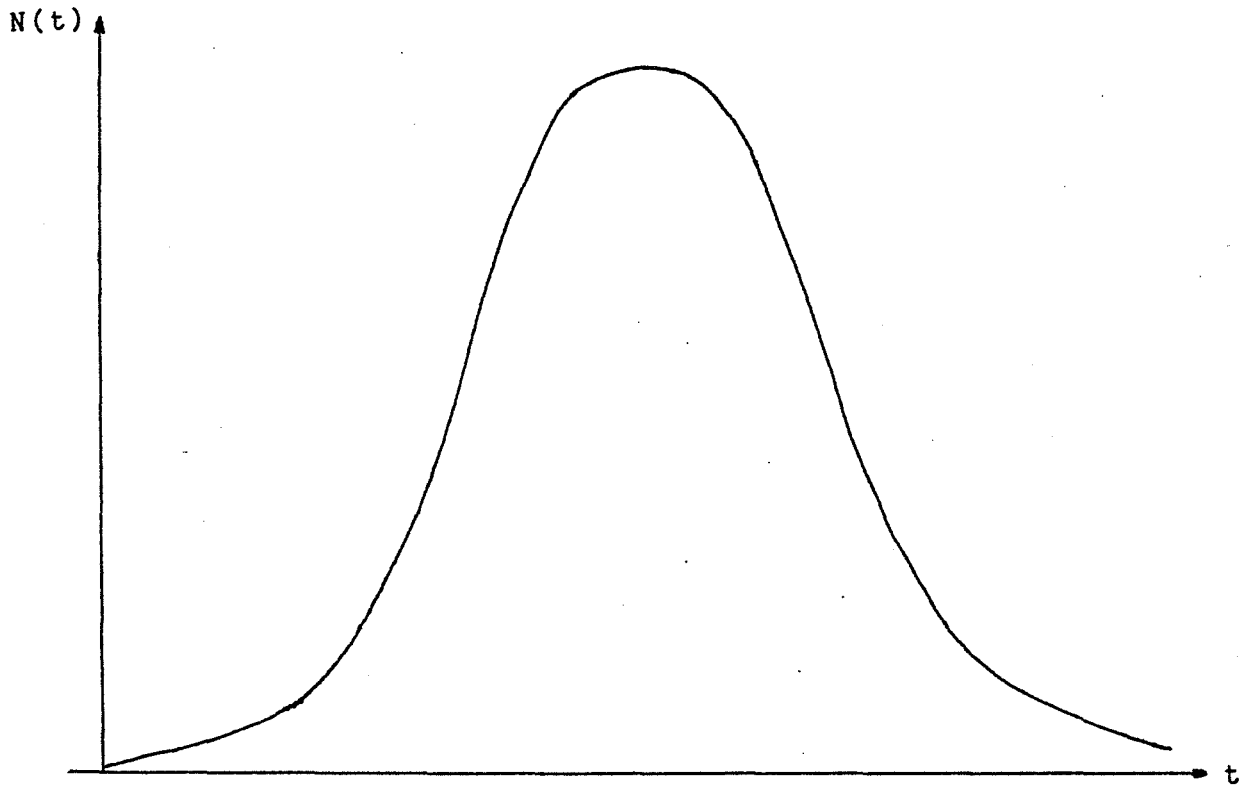
- un point de vue probabiliste qui en fait un phénomène déterministe résultant de l'agrégation d'un grand nombre d'événements aléatoires à un niveau microscopique.

La notion de filtre linéaire si elle est opératoire n'explique que difficilement le rapport entre le délai employé et la réalité perçue par le modélisateur. Il existe peu de système où une structure ou un objet peut être considéré comme un filtre, par contre souvent il y a quantités de petits faits aléatoires qui agrégés donnent une allure déterministe au phénomène. Par exemple prenons le cas où une personne poste un nombre N de lettres pour une même destination. L'expérience quotidienne nous suggère que toutes les lettres en question n'arriveront pas en même temps: au niveau d'une lettre particulière la date d'arrivée sera aléatoire. Si nous considérons par contre l'ensemble des lettres, l'arrivée obéit à une loi bien déterminée et la courbe de répartition des arrivées aura une forme bien précise qui se retrouvera pour toutes les expériences.

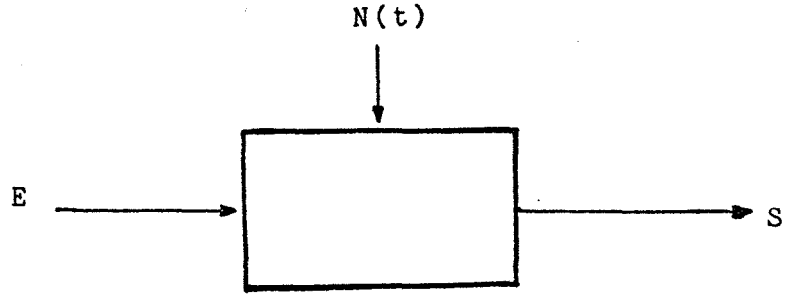
Pour une lettre déterminée nous aurons donc une date d'arrivée aléatoire obéissant à une loi déterminée:



Pour un grand nombre de lettres, nous aurons un phénomène déterministe donnant une répartition des arrivées de la forme suivante:



La vision automatisme dira que ce délai est un filtre linéaire de réponse impulsionnelle N(t)



Il est possible d'expliciter cette réponse impulsionnelle. Pour un délai d'ordre 1 qui comporte donc un seul niveau, nous avons les relations suivantes :

$$\begin{cases} \frac{d\,N(t)}{dt} = IN(t) - OUT(t) \\ \\ OUT(t) = N(t)/DEL \end{cases}$$

soit $\frac{dN(t)}{dt} = IN(t) - N(t)/DEL$

en passant aux transformées de Laplace :

$$s.\mathcal{N}(s) = \mathcal{In}(s) - \mathcal{N}(s)/DEL$$

soit $$\mathcal{N}(s) = \mathcal{In}(s)/(s+1/DEL)$$

Si l'entrée correspond à une impulsion, nous avons $\mathcal{In}(s) = 1$ et donc :

$$\mathcal{N}(t) = e^{-t/DEL}$$

On peut calculer la réponse à un échelon en utilisant la relation classique :

$$\mathcal{Y}(s) = \mathcal{H}(s) \cdot \mathcal{U}(s)$$

avec $\mathcal{Y}$ sortie
$\mathcal{H}$ entrée
$\mathcal{U}$ fonction de transfert

Dans le cas présent, nous avons :

$$\mathcal{V}(s) = 1/(s+1/DEL)$$

$$\mathcal{H}(s) = E/s \qquad E \text{ hauteur de l'échelon}$$

d'où $$\mathcal{N}(s) = (E/s).(1/(s+1/DEL))$$

soit $$N(t) = E.DEL.(1-e^{-t/DEL})$$

ce qui donne pour le flux de sortie $$OUT(t) = E.(1-e^{-t/DEL})$$

De même pour un délai d'ordre n, nous pouvons calculer la réponse impulsionnelle. Rappelons que dans ce cas nous avons n délais d'ordre 1 en cascade et que pour des raisons de cohérence nous appelons DEL le délai total de traversée de ces délais, par conséquent chaque délai d'ordre 1 à un temps de traversée égal à DEL/n.

Nous avons donc le système suivant :

(1) $N1(t) = IN(t) - N1(t).n/DEL$

(2) $N2(t) = N1(t).n/DEL - N2(t).n/DEL$

. .

. .

. .

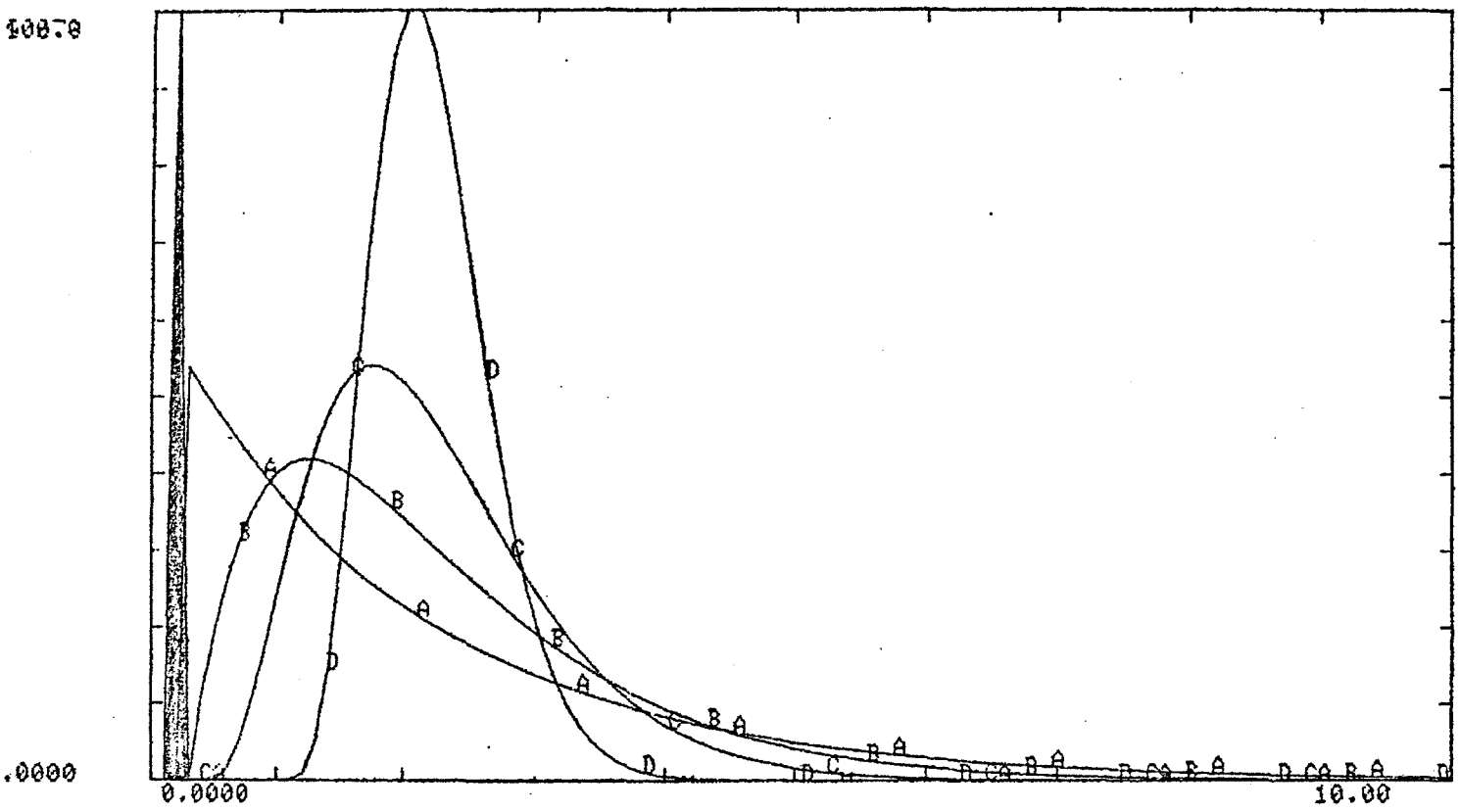
(n) $Nn(t) = Nn-1(t).n/DEL - Nn(t).n/DEL$

Ce qui donne en résolvant chaque équation en cascade :

$$Nn(t) = (n/DEL)^{n-1}.(t^{n-1}/(n-1)!).e^{-n.t/DEL}$$

et pour le flux de sortie $$OUT(t) = (n/DEL)^{n}.(t^{n-1}/(n-1)!).e^{-n.t/DEL}$$

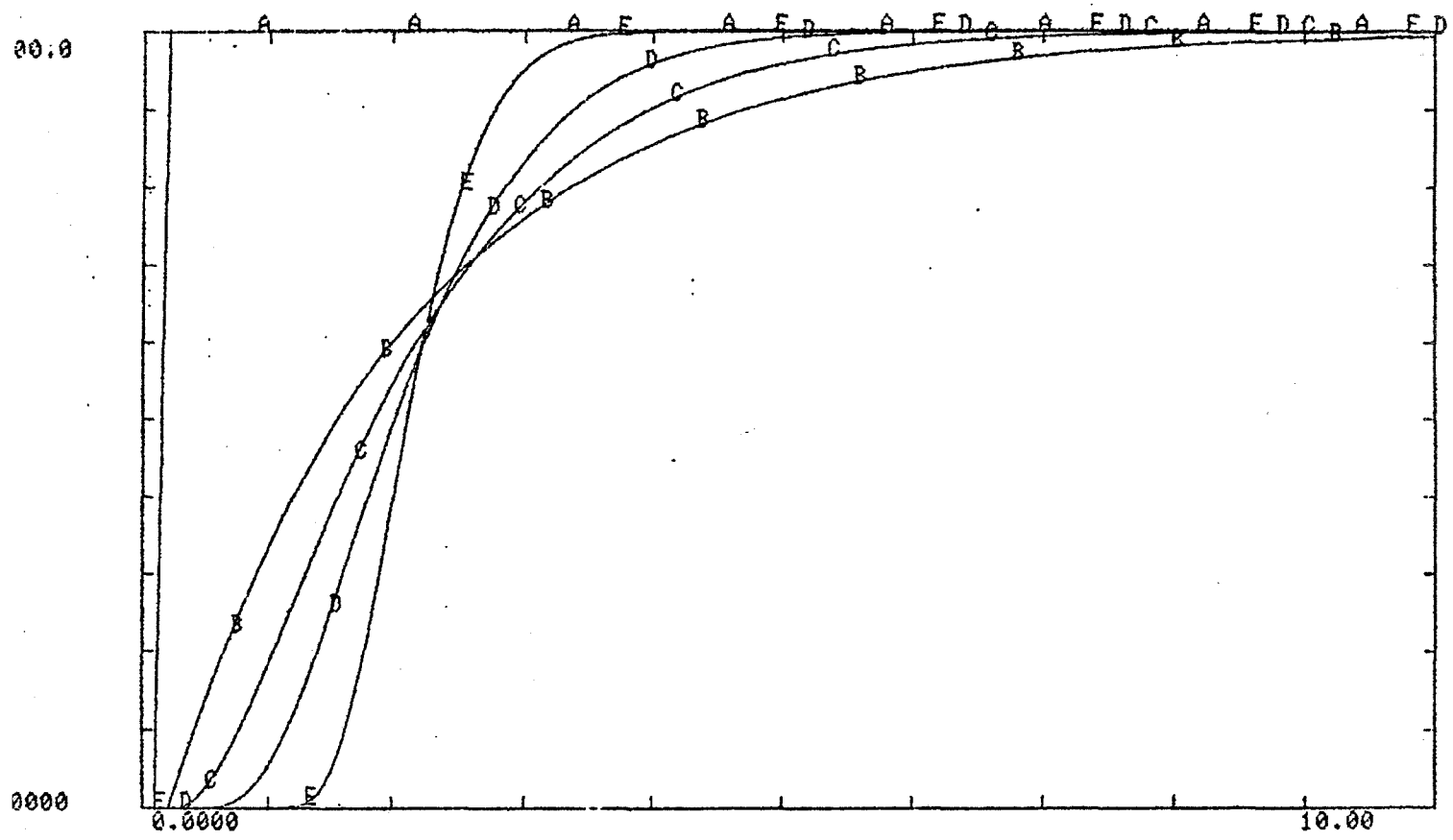
Si nous considérons ce délai comme le composé d'un ensemble d'événements aléatoires, nous pouvons dire que la réponse à une impulsion nous donne la répartition des arrivées dans le temps des quantités élémentaires, en fait l'histogramme de ces arrivées. Par une rapide analogie, nous pouvons considérer que l'expression de OUT(t) est la densité de la loi de probabilité que suivent les événements aléatoires. Ceci permet une remarque intéressante quant à la compréhension du phénomène : un délai d'ordre n est en fait la résultante d'événements aléatoires suivant une loi gamma de paramètres n et n/DEL (pour un délai d'ordre on retombe sur un processus poissonien). Ce qui donne comme espérance (temps de traversée moyen du délai) DEL, ce qui est heureux, et comme variance DEL^2/n. L'expression de la variance permet de remarquer que quand l'ordre devient grand, le délai se rapproche d'un retard pur.



Réponse à une impulsion de hauteur 100

d'un délai d'ordre	1	DELAI = 2	courbe A
.....	2	.....	..... B
.....	4	.....	 C
.....	10	.....	..... D

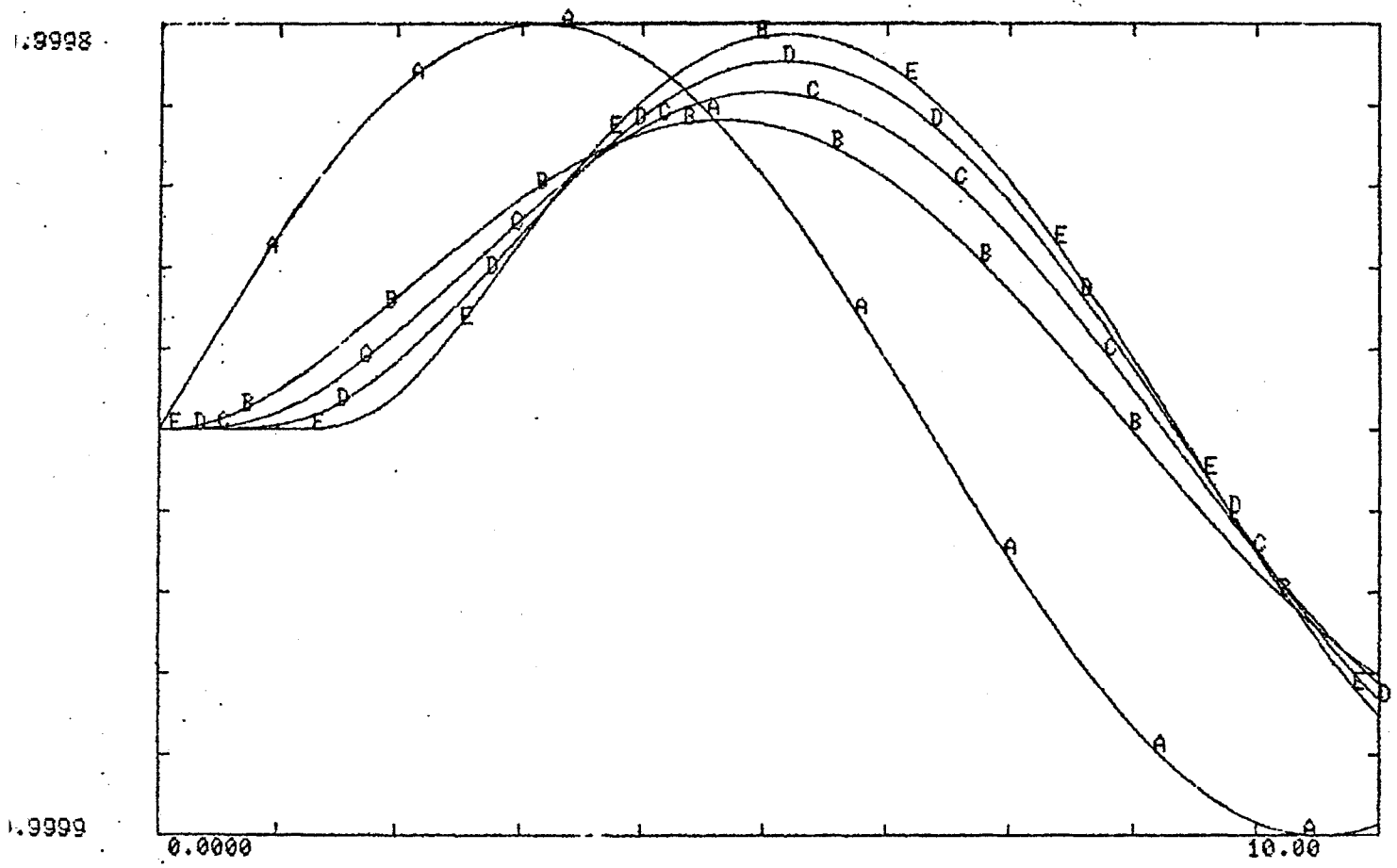
Fig. 5



Réponse à un échelon

délai d'ordre 1	DELAI 2	B
..... 2	.....	C
..... 4	.....	D
..... 10	.....	E

Fig. 6



Réponse à une sinusoïde

Fig. 7

La présence des délais dans un modèle restreint le choix du pas de simulation. En effet, intuitivement il est facile de comprendre que si la valeur du délai est inférieure au plus petit intervalle de temps perceptible par le modèle, c'est à dire dt, il n'y a en fait pour le modèle pas de délai: il n'existe pas. Mathématiquement ce problème est un problème d'échantillonnage et le théorème de NYQUIST indique que la période d'échantillonnage doit être inférieure de moitié à la plus petite période du phénomène observé. Dans notre cas la période d'échantillonnage est dt et le phénomène est le délai, donc dt doit être inférieur à DELAI/2 pour un délai d'ordre 1 soit:

$$dt \leqslant DELAI/(2*\text{ordre du délai})$$

Ceci dit le choix de dt reste délicat: c'est un compromis entre le temps de calcul et la précision des calculs.

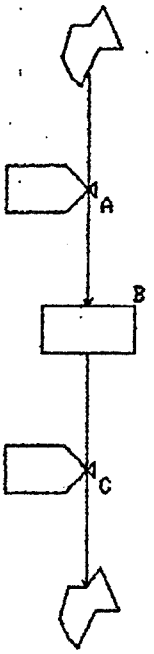
E Le graphisme associé à la Dynamique des Systèmes.

Pour une écriture plus aisée des équations, la dynamique des systèmes offre une représentation graphique des modèles. A chaque type de variable

niveau
taux
auxiliaire

correspond un symbole graphique. Il existe de plus des régles de construction du graphe correspondant aux liaisons entre ces éléments. Ces régles ne sont pas formalisées mais découlent du type des variables et des relations pouvant exister entre ces types.

Un niveau pour exister a besoin de flux d'entrée et de sortie. Ces flux commandés par des vannes (taux) proviennent soit



EXEMPLE FLUX ET NIVEAU

de l'environnement du système représenté par des puits, soit d'autres niveaux. Pour contrôler ces flux, les vannes ont besoin d'information, celle-ci provient soit des niveaux, soit d'autres vannes, soit de variables auxiliaires.

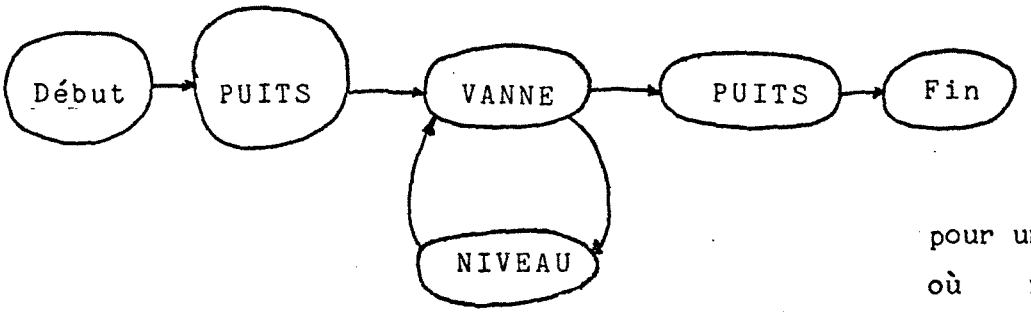
Nous sommes sensibles au fait que de telles régles de construction sont floues et peu pratiques. De plus si nous voulons introduire un contrôle "syntaxique" par une machine, il nous faudra formaliser un peu plus et construire une "grammaire" pour ce "langage graphique".

L'idée de construire des grammaires décrivant des graphes ou des figures structurées n'est pas neuve. Des auteurs tels que SHAW, ROSENFELD, FU ou PFALTZ s'en sont préoccupés dans le cadre de travaux sur la reconnaissance des formes ou sur l'intelligence artificielle. Des grammaires ont été élaborées:

- grammaire de "plexes" (NARASIMHAN et FEDER)
- grammaire de graphe (web grammar de PFALTZ et ROSENFELD)

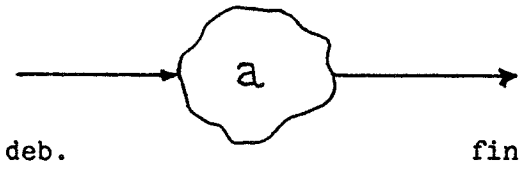
(voir exemples)

Ces grammaires permettent de construire des réseaux complexes. Elles permettent donc de formaliser globalement ces structures. Pour notre propos, il n'est pas nécessaire d'étudier le problème dans son ensemble, il suffira de determiner si tous les chemins du graphe sont conformes à la grammaire. Cette grammaire a une propriété intéressante: le symbole courant dépend directement du symbole précédent et de lui seul. Un chemin correct du graphe pourra donc être décrit par un automate à état fini relativement simple:



pour un réseau où n'interviennent que les liaisons

Exemple de grammaire de plexe (SHAW)

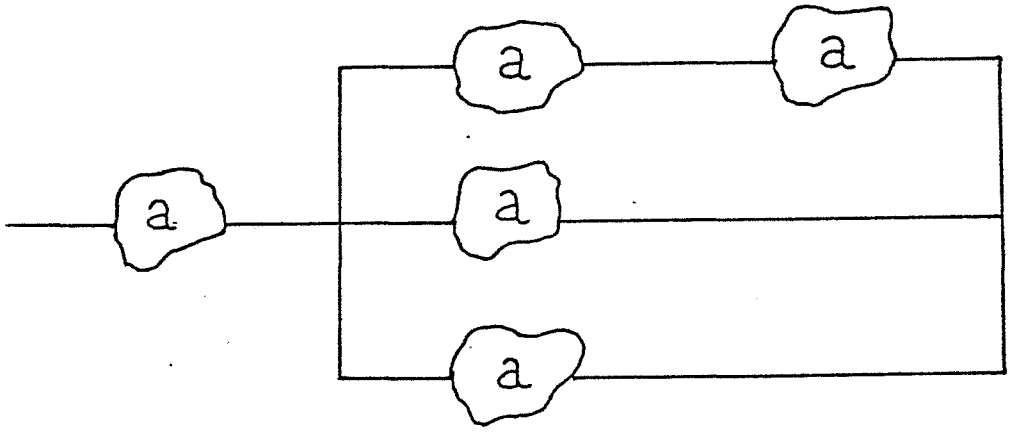


objet primitif a

S ⟶ a | (S*S) | (S+S) grammaire

Cette grammaire signifie que S peut être dérivé en a ou en un couple (S,S) en parallèle ou en série.

La structure suivante vérifie cette grammaire:



ce qui correspond à (a+(((a+a)*a)*a))

Grammaire de graphe (PFALTZ et ROSENFELD)

éléments primitifs: $\left\{ t1 \longrightarrow t2,\ t1 \xrightarrow{A} t2 \right\}$

Grammaire:

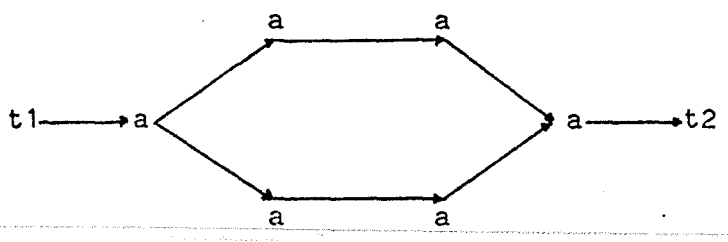
1) $\overset{A}{\bullet} := \overset{A1}{\bullet} \longrightarrow \overset{A2}{\bullet}$; $E = \left\{ (P,A1) \mid (P,A) \text{ une liaison dans graphe précédent} \right\} \cup \left\{ (A2,Q) \mid (A,Q) \text{ une liaison dans graphe précédent} \right\}$

Ce qui signifie que pour remplacer A par $\overset{A1}{\bullet} \longrightarrow \overset{A2}{\bullet}$, il faut lier A1 au graphe existant par les liaisons $P \longrightarrow A$ existantes, et il faut lier A2 au graphe existant par les liaisons $A \longrightarrow Q$ existantes.

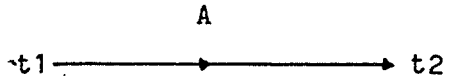
2) $\overset{A}{\bullet} := \overset{\bullet A}{\bullet}$ si $\exists!$ P et Q | les liaisons (P,A) et (A,Q) existent $E = \left\{ (P,A) \mid (P,A) \right\} \cup \left\{ (A,Q) \mid (A,Q) \right\}$

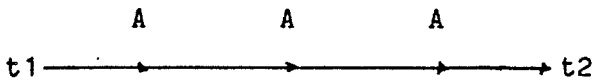
3) $\overset{A}{\bullet} := \overset{a}{\bullet}$; E= précédentes régles

Ces régles permettent d'obtenir la structure suivante:

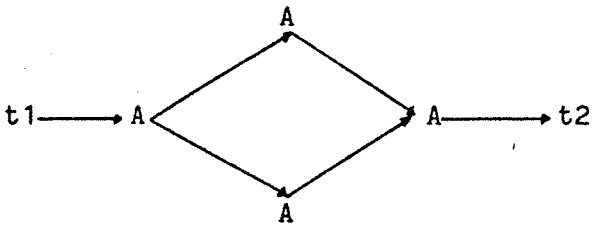


La structure précédente est obtenue par les transformations suivantes:

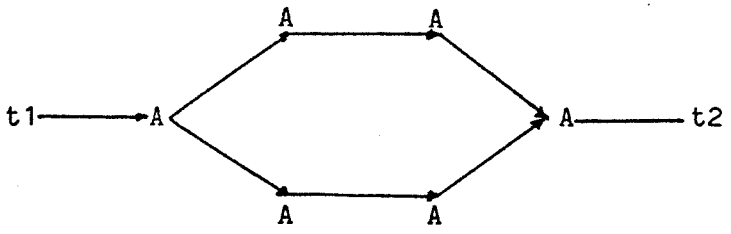




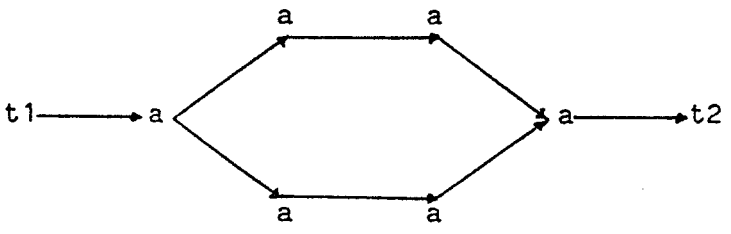
par application de la régle 1



par application de la régle 2

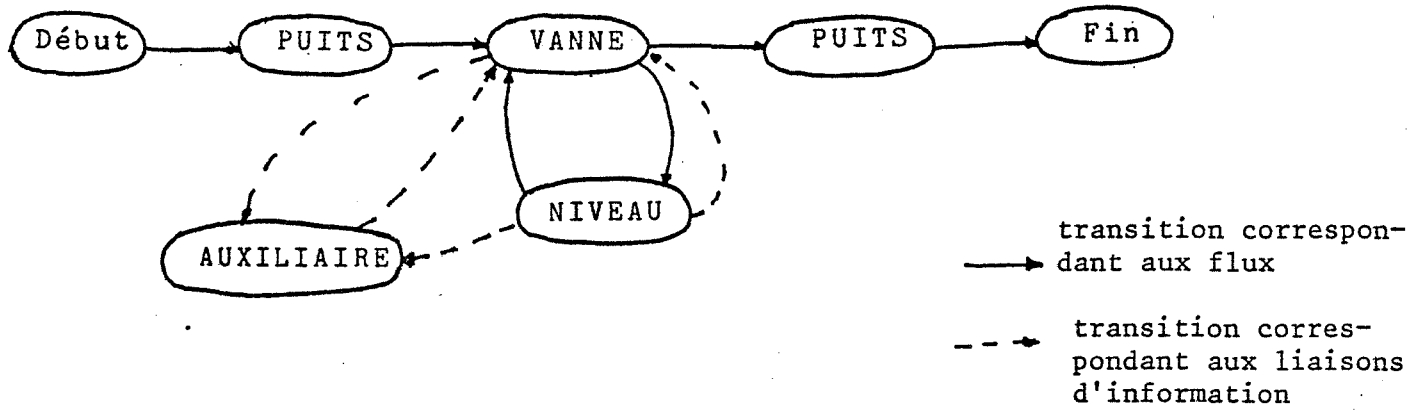


par application de la régle 1

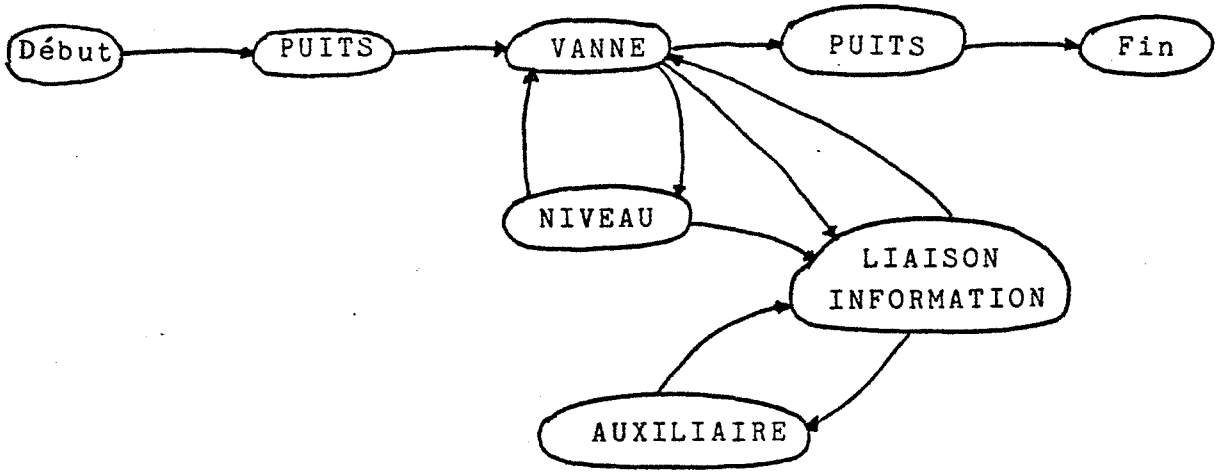


par application de la régle 3

Pour un réseau complet, il faut soit deux automates différents, soit un automate avec deux types de transition, soit introduire un nouvel élément à part entière la liaison information.



Automate à deux types de transition



Automate avec liaison d'information parmi les états

Pour notre système, nous avons utilisé le dernier type d'automate. De plus, le fait que les différents symboles valides pour la succession du symbole courant ne dépendent que de ce symbole et non pas de "l'historique du graphe", nous autorise à ne prendre en compte que les transitions indépendamment de la structure complète du graphe. Ce qui fait que nous utilisons la matrice des successeurs suivante:

	puits	vanne	niveau	auxil.	l. info.
puits		X			
vanne	X		X		X
niveau		X			X
auxil.					X
l. info.		X		X	

du fait des contraintes physiques qui s'appliquent à la construction du graphe, nous sommes revenus à une grammaire normale. La construction ne pouvant être qu'un processus linéaire, elle peut se représenter par une chaine du type:

puits 1, vanne 1, niveau 1, vanne 2, puits 2, placer le "contexte" en niveau 1, vanne 3, niveau 2, vanne 4, puits 3 ....

La grammaire analyse en fait cette chaîne et non le graphe lui-même. Le problème d'une grammaire de graphe se reposerait si nous voulions analyser la correction syntaxique d'un graphe déjà construit. Pour ce cas une solution intellectuellement satisfaisante mais délicate d'implémentation serait d'utiliser des processus d'exploration parallèle suivant un algorithme du type:

1) rechercher tous les puits "départ" d'un réseau.
2) rechercher toutes les auxiliaires de type INPUT
3) de chacun des sommets répertoriés précédemment faire naitre un processus qui descend les chaines en vérifiant la syntaxe grâce aux automates vus plus haut.

La descente des chaines se ferait ainsi:

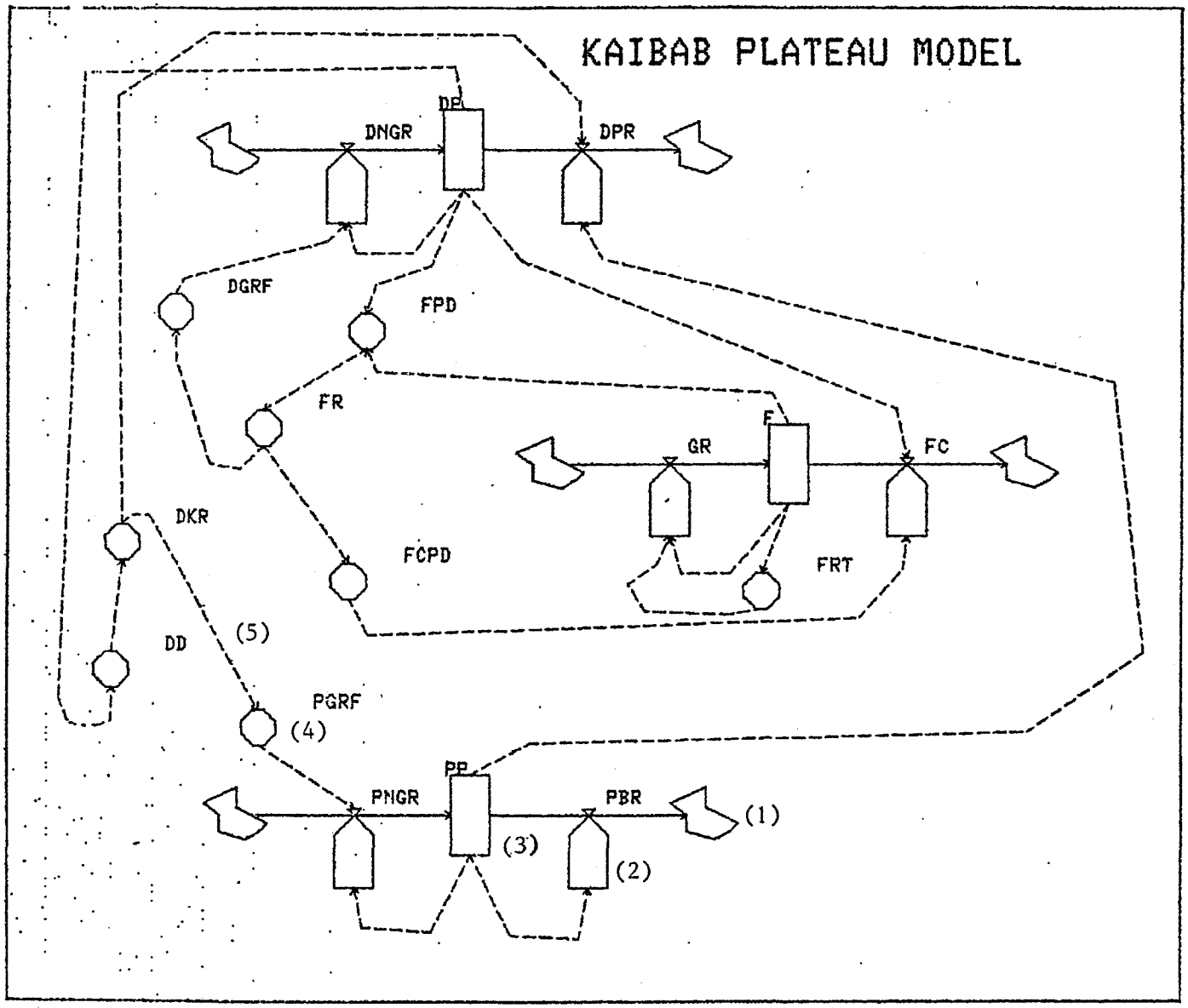
- arrivé à un puits le processus meurt

- arrivé à une bifurcation du type  le processus fait naitre autant de processus qu'il y de branches nouvelles.

- arrivé à un embranchement du type  le processus vérifie qu'il y a autant de processus en attente que de branches incidentes autres que la sienne, si oui il tue ces processus et continue, sinon il se met en attente.

- arrivé à une auxiliaire sans liaison issue de cette auxiliaire le processus meurt.

Un tel algorithme est simple mais mal adapté à la structure des machines informatiques actuelles. Pour l'employer il faut transformer les naissances et les morts de processus en marquages de sommets et en effaçages de ces marques, ce qui est nettement moins séduisant.



Exemple de réseau complet comportant

des puits (1)
des vannes (2)
des niveaux (3)
des auxiliaires (4)
des liaisons d'information (5)

F. Les applications

La dynamique des systèmes, nous l'avons vu, permet de modéliser des systèmes en les représentant par des systèmes canoniques d'équations différentielles du premier ordre. Cela donne des systèmes d'équations du type:

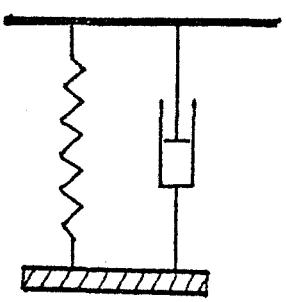
$$\frac{dX}{dt} = f(.....)$$

et

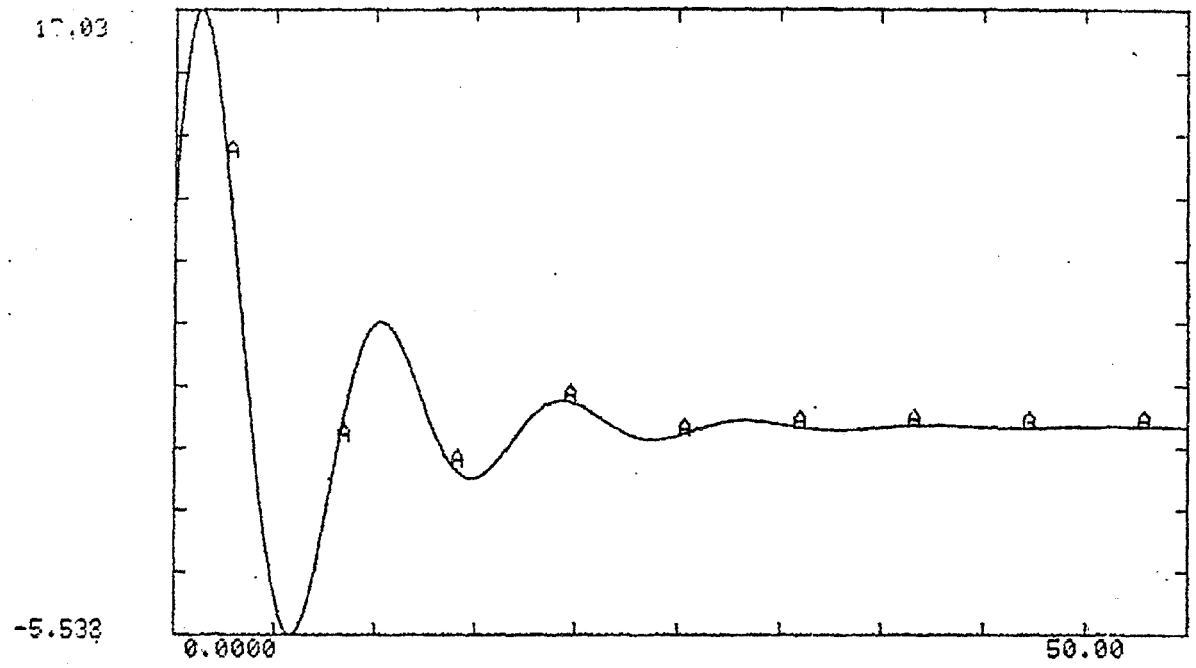
$$Y(t) = g(....)$$

La souplesse d'emploi de tels systèmes permet de modéliser de façon "grossière" les grandes relations entre les variables importantes d'un système ou au contraire de descendre à un niveau de représentation très fine. Cette latitude dans le choix du degré de précision de la représentation fait que ce type de modèlisation est très bien adapté à l'étude du comportement des grands systèmes sociaux économiques. Mais ce n'est pas là la seule application possible bien qu'on l'oublie souvent.

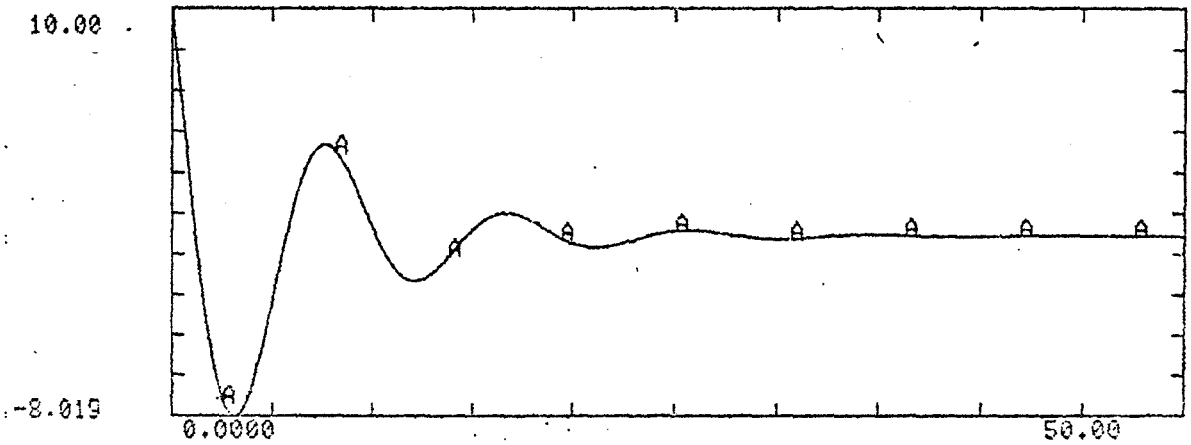
Les modèles étant représentés par des systèmes d'équations différentielles, il est facile d'étudier ainsi des systèmes physiques du type masse ressort avec amortissement. L'intérêt dans ce cas est que tous les types de forces imaginables peuvent être testés.(forme de f(t) dans les équations)



$$m\ddot{x}+d\dot{x}+kx=f(t)$$



A X MIN -5.538 MAX 17.03



A Y MIN -8.019 MAX 10.00

$$m\ddot{x} + v\dot{x} + kx = f(t)$$

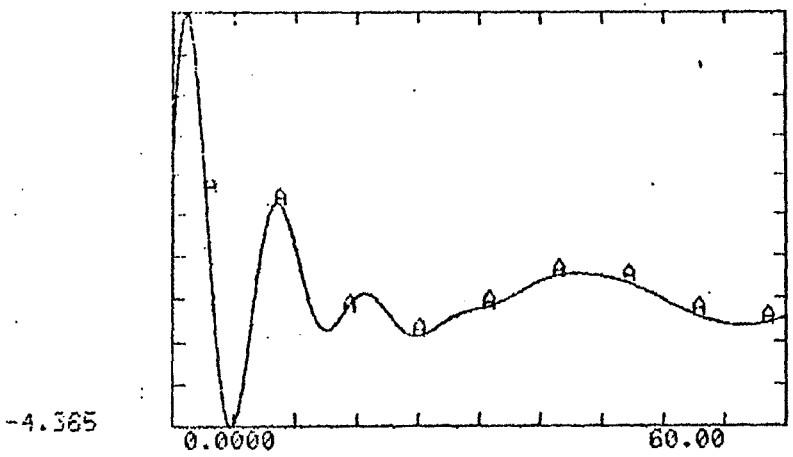
Dans ce cas f(t) est une constante.

Le modèle de dynamique des systèmes correspondant est:

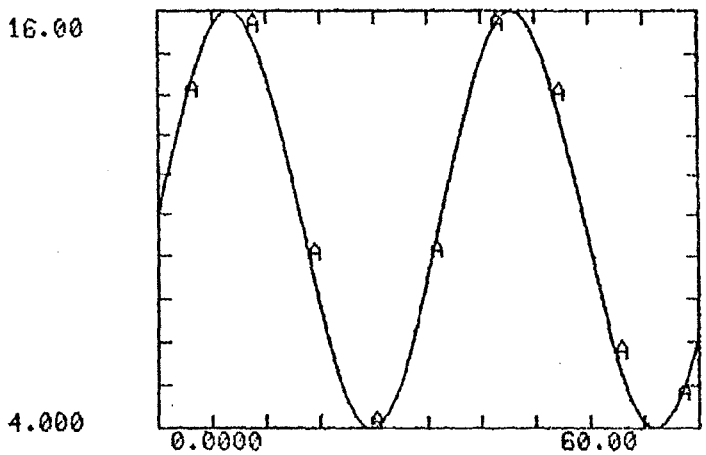
```
X(K) = X(J) + DT*Y(J)                          X élongation
Y(K) = Y(J) + DT/m*(f(t)-kX(J)-vY(J))          Y vitesse
```

$m\ddot{x} + v\dot{x} + kx = f(t)$

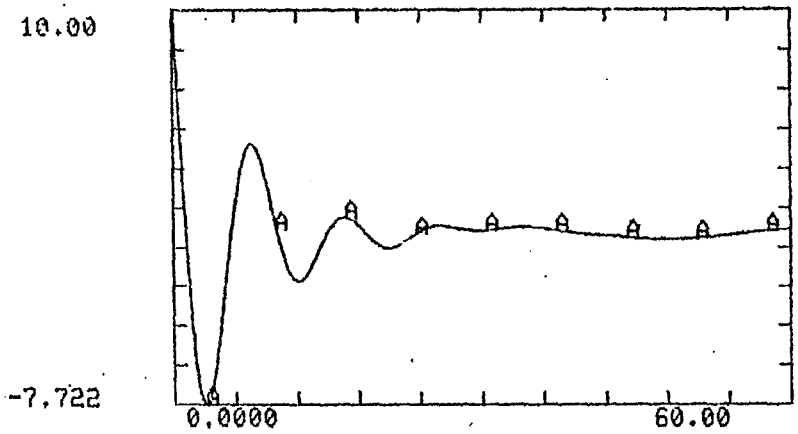
Dans ce cas $f(t) = a \sin(bt)$



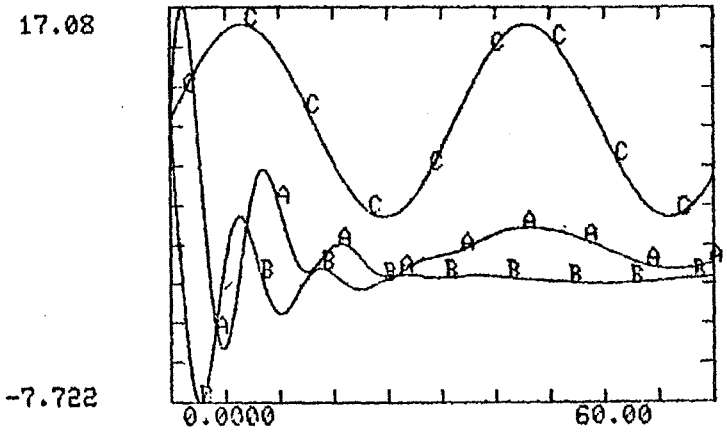
A X MIN -4.365 MAX 17.08



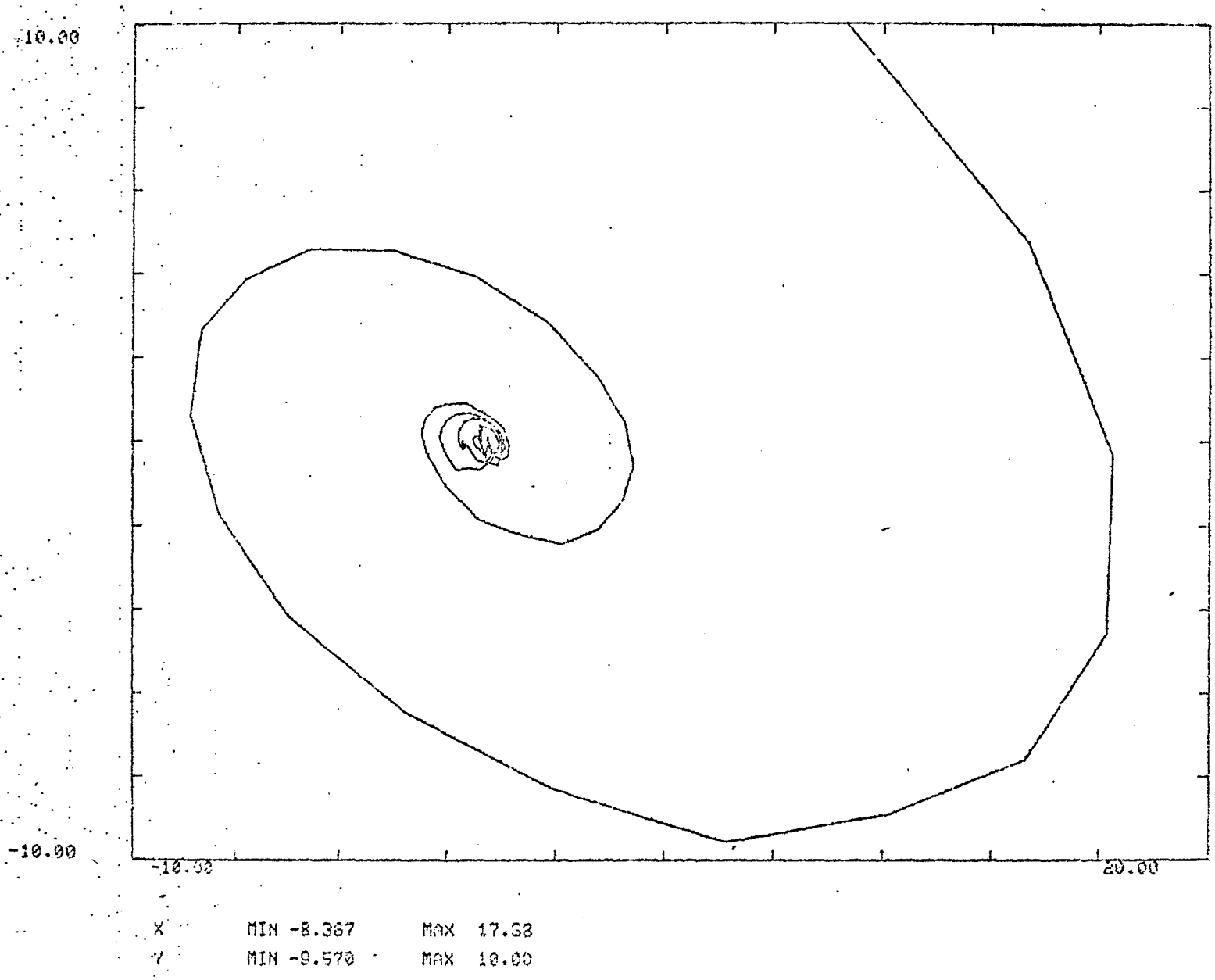
A F MIN 4.000 MAX 16.00



A Y MIN -7.722 MAX 10.00

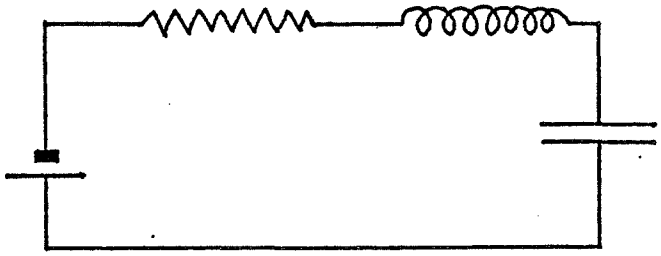


A	X	MIN -4.365	MAX 17.08
B	Y	MIN -7.722	MAX 10.00
C	F	MIN 4.000	MAX 16.00



Représentation de la variation de Y (vitesse) en fonction de celle de X (élongation).

et les analogues électriques RLC:



Ces exemples sont simples mais on peut rapidement les compliquer à l'envie.

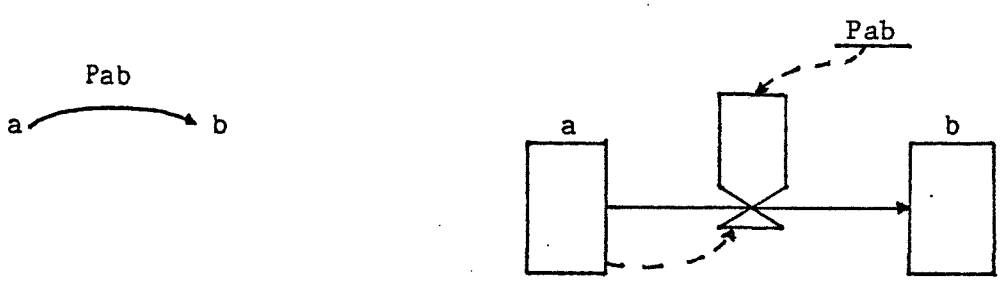
Mais l'application privilégiée reste l'entreprise industrielle et les nombreux problèmes qui peuvent y apparaître. En fait la dynamique des systèmes permet d'étudier rapidement (un modèle complexe comporte une centaine de variables) des comportements qui semblent pathologiques en vérifiant des hypothèses sur les causes de ces comportements. Les applications sont donc nombreuses:

- problème de gestion de stock
- de stratègie à court et long terme
- politique des ventes
- gestion du personnel
- contrôle de gestion

Le principal inconvénient, qu'il ne faut jamais perdre de vue, est que les résultats chiffrés ne sont souvent pas dignes de confiance. Il vaut mieux considérer les résultats comme des indications de tendances plutôt que comme des prévisions chiffrées. En effet le peu de variables et le flou dans les relations entre ces variables du à la difficulté d'estimation des paramètres ne permettent pas une grande précision dans les calculs. Ce défaut est quand même largement compensé par les qualités de ce type de modélisation si on n'attend que des indications sur le type de comportement du système.

Le domaine où tous les avantages de la dynamique des systèmes sont mis en valeur reste les grands systèmes complexes de tout type (sociaux économiques, écologiques, urbanisation etc...). Comme les prévisions chiffrées ne sont en général pas nécessaires, mais que seules les tendances sont utiles, le défaut signalé plus haut perd de son importance. Pour ce type d'application, citons notamment le modèle du monde de FORRESTER puis de MEADOWS, des modèles d'aménagement urbain, des études de cours des matières premières. La souplesse d'utilisation prend ici toute sa valeur: on peut tester rapidement une hypothèse, la modifier ou en changer complètement et ce en modifiant simplement quelques équations ou quelques paramètres.

En poussant cette technique de modèlisation dans ses retranchements on peut étudier d'autres classes de modèles. Une classe intéressante et facile à transposer est celle des modèles markoviens ou semi-markoviens. La transformation est facile à effectuer et le modélisateur a tout à y gagner: l'étude est simplifiée grâce aux outils fournis et à la faculté d'étudier les régimes transitoires. L'équivalence se fait de la façon suivante



Au niveau des équations la transformation est facile. Pour un processus stationnaire nous avons:

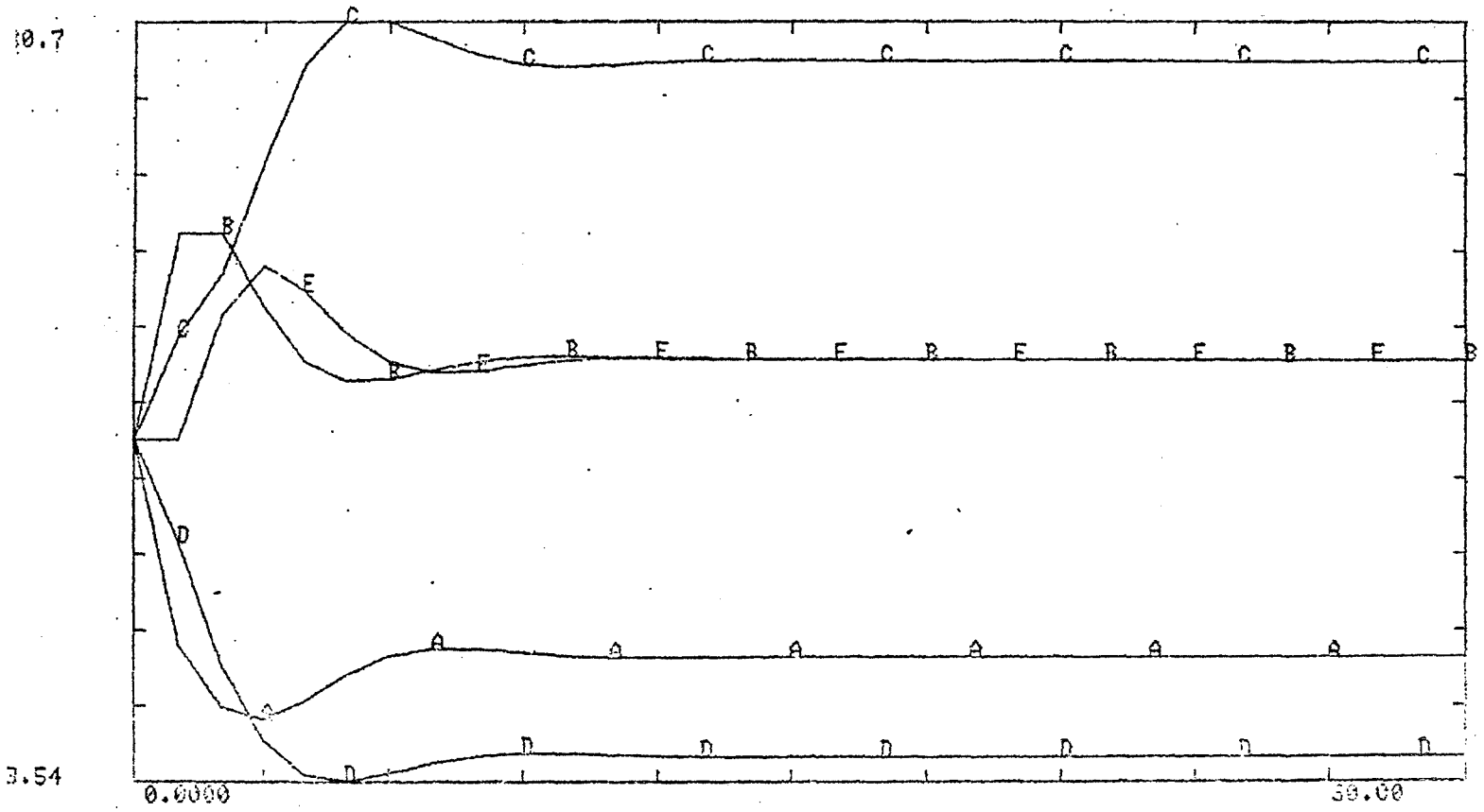
$$\Pi_{j(n)} = \sum_i \Pi_{i(n-1)} P_{ij}$$

$\Pi_{j(n)}$ étant la probabilité d'être dans l'état j à l'instant n.

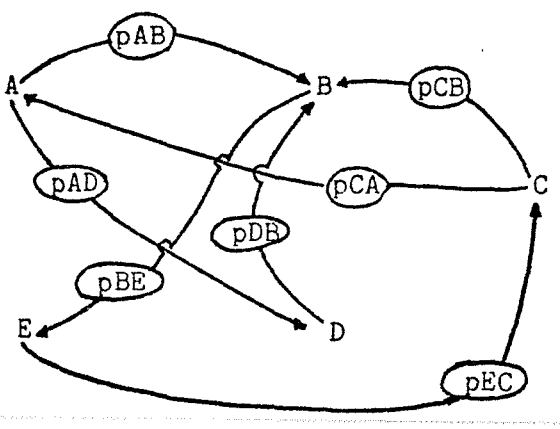
```
LISTING  EQUATIONS DU MODELE

A(K)=PCA*C(J)-(PAB+PAD)*A(J)+A(J)
A=100
B(K)=PAB*A(J)+PDB*D(J)+PCB*C(J)-PBE*B(J)+B(J)
B=100
C(K)=PEC*E(J)-(PCA+PCB)*C(J)+C(J)
C=100
D(K)=PAD*A(J)-PDB*D(J)+D(J)
D=100
E(K)=PBE*B(J)-PEC*E(J)+E(J)
E=100

POUR LA SUITE TAPER  CR
```



Modèle markovien élémentaire.



On peut écrire:

$$\Pi_{j(n)} = \Pi_{j(n-1)} P_{jj} + \sum_{i \neq j} \Pi_{i(n-1)} P_{ij}$$

or si on remarque que $\sum_{k} P_{jk} = 1$ et que donc $P_{jj} = 1 - \sum_{k \neq j} P_{jk}$, on a la forme d'équation:

$$\Pi_{j(n)} = \Pi_{j(n-1)} + \sum_{i \neq j} \Pi_{i(n-1)} P_{ij} - \sum_{k \neq j} \Pi_{j(n-1)} P_{jk}$$

ce qui permet de faire une analogie entre les équations de niveau d'un système dynamique et cette équation.

Les Π_j seront convertis en niveaux et les flux seront les quantités transitant d'un état à un autre par unité de temps, soit $\Pi_{i} P_{ij}/dt$.

Les limites de la dynamique des systèmes sont faciles à définir. Elle s'intéresse à des modèles fortement agrégés. En effet pour parler de flux, donc de quantités divisibles aussi finement qu'on le désire, il faut s'intéresser à des grands nombres face aux "individus" concernés. La limite d'application sera donc précisée par le niveau d'abstraction ou d'agrégation du modèle:

- s'interesse-t'on au comportement du modèle indépendamment des entités qui traversent ou circulent dans le système: la dynamique des systèmes pourra en général s'appliquer.

- s'interesse-t'on au comportement et au devenir des entités qui circulent dans le système, c'est à dire s'interesse t'on à un niveau microscopique, alors dans ce cas la dynamique des systèmes ne pourra pas s'appliquer ou alors difficilement avec forces astuces. Il vaut mieux choisir un autre type de simulation (simulation discrète par exemple).

III DYNAMINE GRAPHIQUE : structure et fonctionnement.

La construction, la mise au point puis l'utilisation d'un modéle suivent une procédure en plusieurs étapes où les retours en arrière sont fréquents et nécessaires et d'où un certain parallélisme n'est pas absent.

Pour qu'un système d'aide à la modélisation soit utile, il est nécessaire qu'il permette de suivre cette procédure dans une large mesure et d'opérer sans peine les modifications et les remises en cause du modèle.

Nous allons voir quelles reflexions ont présidé à l'élaboration de DYNAMINE puis dans quelle mesure ce système peut répondre aux besoins du modélisateur.

Le processus de modélisation est une activité nécessitant de nombreuses compétences. Cette activité interdisciplinaire demande un dialogue constant entre les différents membres de l'équipe.

Les membres de cette équipe peuvent être affectés à l'un des deux groupes suivants:

- le groupe spécialiste du ou des domaines se rapportant au système à modéliser.

- le groupe "informatique" qui regroupe toutes les personnes s'occupant de la technique même de modélisation (essentiellement programmation et résolution des problèmes techniques dus à la machine)

Il va de soi que ce qui peut arriver de mieux est que ces deux groupes soient confondus, c'est à dire que les spécialistes non informaticiens soient capables de prendre en charge tout le côté technique de la modélisation sans toutefois être submergés par cet aspect mineur. Il est donc souhaitable d'élaborer des systèmes informatiques qui permettent d'obtenir des résultats sans trop de peines, rapidement tout en ne nécessitant pas trop de connaissances informatique. Ces systèmes doivent pouvoir donner les informations nécessaires à la poursuite des opérations, en guidant l'utilisateur et en "pardonnant" les erreurs inévitables dans un tel processus.

Pour notre part nous avons appliqué ces considérations à la Dynamique des Systèmes, cette technique de modélisation étant d'un emploi relativement simple tout en conservant de larges possibilités, de plus il existe un symbolisme graphique pour elaborer les modèles qualitativement avant d'écrire les équations.

Nous avons mis au point un système d'aide à la modélisation qui tient compte non seulement du langage mais aussi du processus de modèlisation et de ses diverses étapes. Celles ci peuvent être définies comme suit:

- définition des limites du système

- définition des variables représentant ce système

- définition des relations entre ces variables: construction du diagramme causal.

- construction du diagramme de flux: schéma du modèle en quelque sorte

- écriture des équations

- vérification du modèle. Etape que l'on peut diviser en deux:
  - .vérification syntaxique
  - .vérification sémantique: est ce que le modèle correspond bien à ce qu'on désire ?

- validation

- exploitation

Mais ce découpage reste tout à fait théorique. Les différentes étapes ne sont pas aussi distinctes et souvent se chevauchent ou se confondent. De plus les retours en arrière sont constants et nombreux.

Pour situer le problème, nous allons étudier les deux premières étapes :

- définition des limites

- définition des variables

Ces deux opérations se confondent souvent, car s'il est facile de définir les limites de façon qualitative et générale, c'est à dire de définir les aspects du système étudié qui seront intégrés au modèle, il est plus difficile de définir de façon précise les limites physiques ou temporelles et encore plus délicat d'apprécier à quel niveau d'agrégation et de simplification il faudra se tenir. Pour cela il est nécessaire d'avoir une idée au moins succinte des informations qu'on utilisera tant en données statistiques qu'en relations et équations. Ce qui signifie qu'on est obligé de définir une bonne partie des variables qui représenteront le système étudié pour délimiter le modèle. Ce phénomène d'anticipation se renouvellera pour chaque étape.

D'autre part il n'est pas rare de s'apercevoir dans une étape suivante qu'il est utile d'introduire d'autres variables ou d'autres limites qui permettront de rendre compte de certaines relations et de certains phénomènes non prévus. On s'aperçoit alors que la modélisation loin d'être une activité ordonnée, séquentielle, est plutôt une activité buissonnante qui comporte bien les étapes citées précedemment mais avec un haut parallélisme dans le cheminement et de nombreuses boucles de rétroaction (ce qui est après tout normal pour la Dynamique des Systèmes).

La reflexion qui permet de concevoir un système d'aide à la modèlisation doit donc se porter sur d'autres aspects :

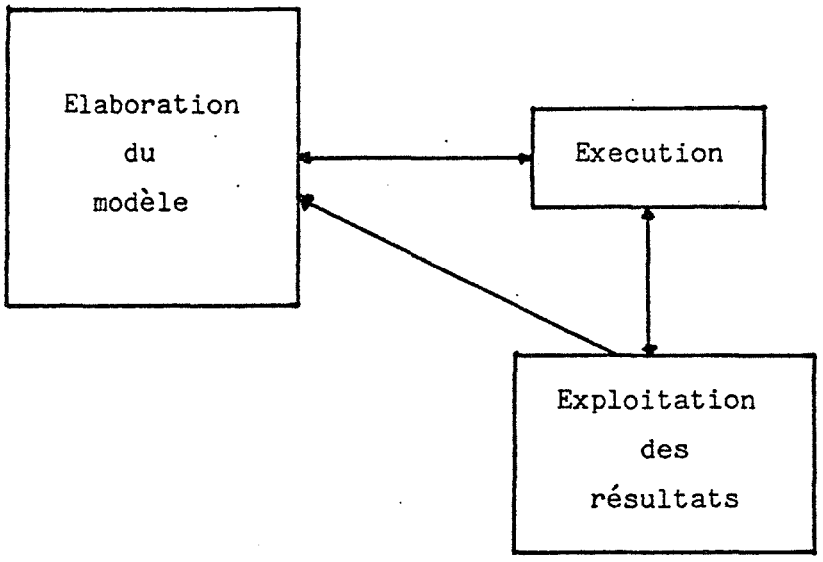
- un découpage moins fin du processus

- une définition de ce qui est du ressort de la reflexion et de la prise de décision du

modélisateur et de ce qui est du ressort de la machine pour décharger l'utilisateur de taches fastidieuses.

A.1 Un découpage plus grossier.

Le découpage se fera de façon moins fine mais plus efficace en ce qui nous concerne. Il y aura une étape d'élaboration du modèle qui par l'intermédiaire de l'exécution aboutira à l'étape d'exploitation des résultats soit pour les tests de validation soit pour l'exploitation du modèle.



Le bloc élaboration comprenant toutes les étapes jusqu'à la vérification. Le bloc résultat permettant la validation et l'utilisation du modèle une fois sa mise au point effectuée.

A.2 un automatisme partiel

Il semble évident qu'un système d'aide à la modélisation ne prendra pas en compte toutes les étapes citées plus haut mais seulement celles qu'il est possible d'effectuer de façon quasi automatique du fait des renseignements fournis par l'utilisateur. Nous devons donc faire la part entre ce que le modélisateur devra encore faire seul face à sa feuille de papier avec un crayon et sa matière grise et ce que fera la machine avec sa non-intelligence mais ses possibilités de calculs et de stockage de données.

L'inventaire de ce que peut faire une machine étant le plus court, nous l'utiliserons pour indiquer ce que devra faire l'utilisateur:

. à partir du diagramme causal fourni par l'utilisateur, moyennant quelques renseignements supplémentaires, déduire le type des variables et donc le diagramme de flux.

. aide au dessin du diagramme de flux.

. déduire certaines équations du modèle du type des variables et des relations existantes (équations de niveaux).

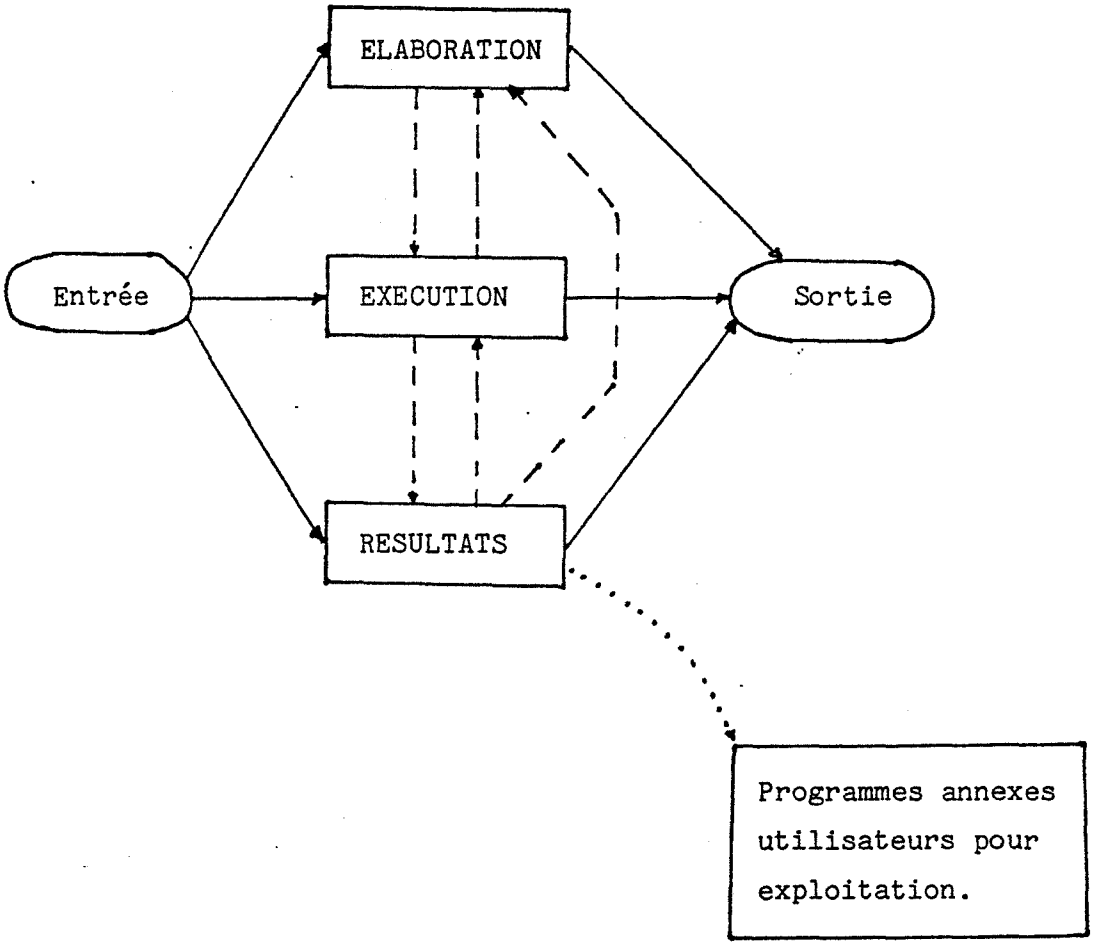
. effectuer toutes les vérifications de type syntaxique et indiquer les erreurs.

. exécuter les simulations demandées en fournissant une aide pour les recherches d'erreurs.

. aider à l'exploitation en mettant à la disposition de l'utilisateur des moyens de visualisation de ces résultats.

. aider à la validation du modèle en fournissant des moyens de tests et un accès facile aux résultats éventuellement par d'autres logiciels que le système.

Le système Dynamine prend en compte la plupart de ces points. De plus sa structure permet à l'utilisateur de mener son travail sans contraintes, les points d'entrée étant nombreux.



Structure générale de Dynamine graphique.

D'une manière générale, les choix se présentent à l'utilisateur sous forme de "menus" qui donnent les renseignements nécessaires. L'usage du clavier en est d'autant limité, ce qui semble être un soulagement pour les utilisateurs non virtuoses de l'AZERTYUIOP.

Les retours en arrière, les modifications sont permis à tous les niveaux dans une large mesure. Les informations ne doivent être fournies qu'une seule fois par l'utilisateur et sont restituées par la machine chaque fois que cela est nécessaire et possible. Dynamine tient donc à jour une masse de données volumineuse et la cohérence des informations a été un des points les plus difficiles à résoudre. Cette cohérence étant encore rendue plus difficile par le nombre des points d'entrée et par les différentes procédures possibles pour construire les modèles.

Nous allons voir maintenant ce qu'il est possible de faire à chaque étape et comment un certain nombre de problèmes a été résolu.

B. Elaboration du modèle.

Dans les systèmes courants, l'utilisateur doit écrire ses équations en utilisant un éditeur de texte non spécialisé et fournir le programme écrit à un compilateur de type DYNAMO. Cette méthode a divers inconvénients. Elle nécessite de la part de l'utilisateur une connaissance du système informatique sous lequel il travaille. L'utilisation d'un éditeur non spécialisé oblige à reporter sur le compilateur toutes les recherches d'erreur ce qui implique une navette entre ce compilateur et l'éditeur et de plus ne permet pas une aide à l'utilisateur au moment où il écrit ses équations (fonctions existantes, paramètres de ces fonctions etc.)

Dynamine comporte un éditeur spécialisé intégré et autorise trois méthodes différentes pour l'élaboration du modèle:

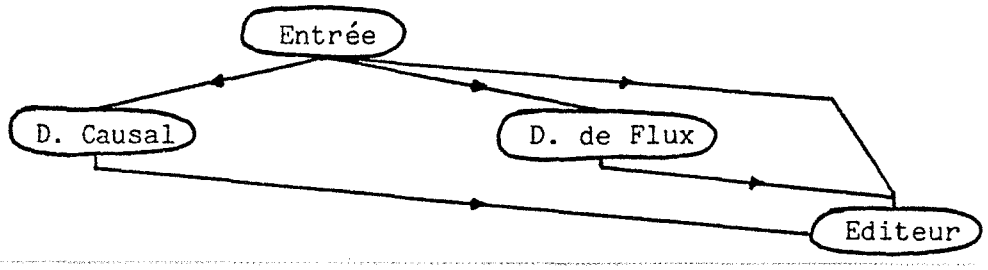
- écrire les équations avec cet éditeur spécialisé

- définir le diagramme causal grâce à un système graphique.

- dessiner le diagramme de flux.

La première méthode est en fait un sous-ensemble des deux suivantes, celles-ci se terminant par l'écriture des équations non générées automatiquement. Ces deux dernières présentent l'intérêt de relier les diagrammes aux équations dans un même système et donc de bénéficier d'un surplus d'information qui autorise des vérifications de cohérence du modèle. Certaines équations peuvent ainsi être générées automatiquement.

Les points d'entrée sont donc les suivants:



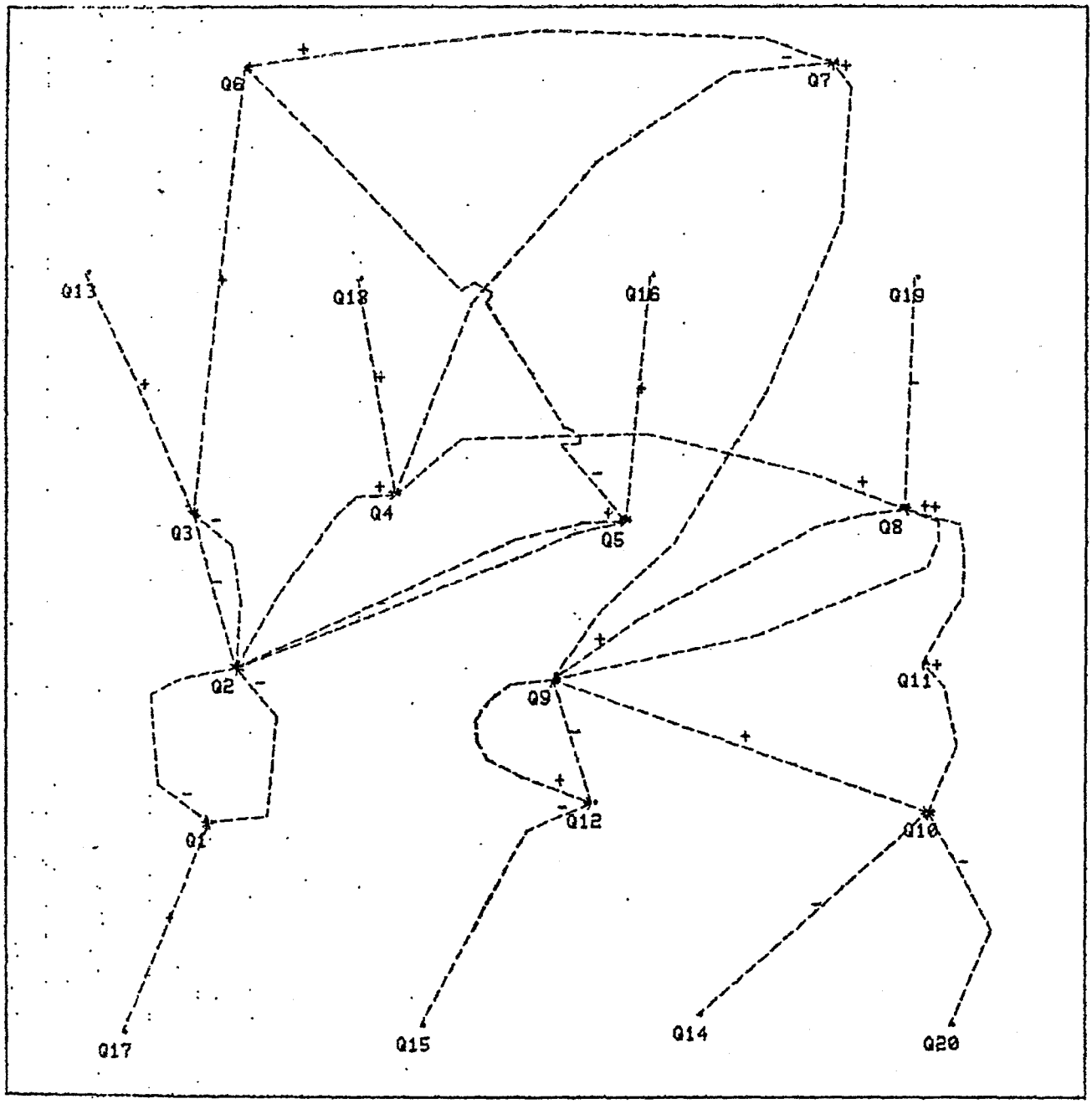
a) Diagramme causal

C'est la méthode qui illustre le mieux le schéma défini plus haut. Le modèlisateur définit d'abord les variables du modèle en fournissant les noms, puis définit les relations entre ces variables en précisant la valeur de la liaison (+ ou -). Ces deux opérations fournissent un graphe de liaisons du modèle sur lequel des opérations d'analyse structurale peuvent être effectuées. Des modules peuvent être incorporés au système qui permettent de se faire une première idée de la structure du modèle et de son comportement futur. De tels modules peuvent être utilisés pour déterminer les variables importantes (en terme de liaisons incidentes ou émises par exemple), les liaisons déterminantes ou les distances entre les variables. Des procédures déterminant le degré de hiérarchisation devraient pouvoir être utilisées de même.

Le diagramme causal est fort utile pour vérifier la cohérence du modèle et la possibilité pour lui de montrer une stabilité dans son comportement grâce à l'analyse des boucles de rétroaction. Cette analyse qualitative du comportement du modèle peut permettre d'économiser des efforts inutiles par la suite. Dans certains cas elle peut même être la seule justifiable, les résultats numériques étant obtenus par une grande agrégation ou pour des grandeurs difficilement quantifiables. Cette remarque s'applique particulièrement aux systèmes complexes sociaux et économiques, champ d'application privilégié de la dynamique des systèmes.

L'étape suivante est le passage du diagramme causal au diagramme des flux, ou plus précisément la reconnaissance à partir du diagramme causal et de sa structure du type des variables du modèle:

niveau
taux
auxiliaires
- Input
- Output
- autres



Exemple de diagramme causal

Cette étape peut être faite de façon semi-automatique sans trop de restrictions sur la structure des modèles. Le choix se situe entre peu de restrictions sur la structure et un degré d'automatisme élevé. Nous allons voir deux méthodes ayant adopté des compromis différents et qui sont exemplaires de la marge de manoeuvre possible. La première est due à BURNS qui en a donné l'algorithme dans un article paru en mars 1979 dans I.E.E.E Transaction. Il se caractérise par un effort important de rationnalisation des définitions et par une axiomatique précise. La deuxième a été utilisée dans Dynamine et demande une intervention de l'utilisateur plus importante dans certains cas, mais implique moins de restrictions.

. Algorithme de BURNS

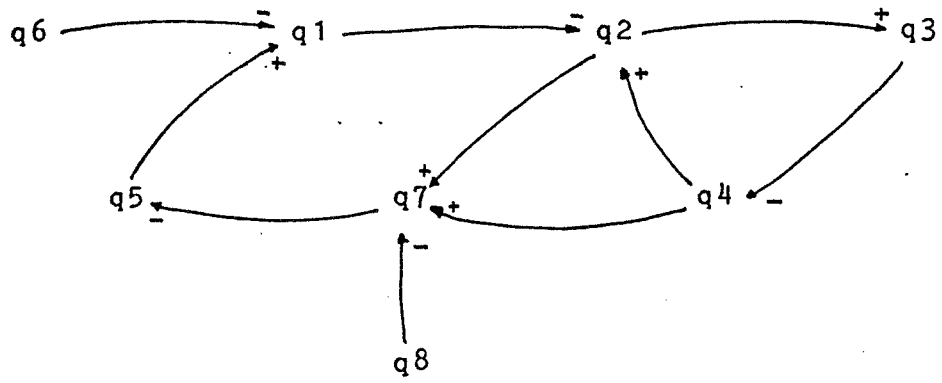
BURNS utilise les notations de la table 1 pour définir les différents ensembles qui permettront de classer les variables. Ces ensembles sont essentiellement les niveaux, les taux, les auxiliaires, les output, les input (divisés en deux paramétres controlables et non controlables). Pour les axiomes et théorèmes il introduit de plus les notions d'affecteur et d'effecteur. Les premiers sont les variables influençant une variable donnée, les suivants sont les variables influencées par une variable donnée. On peut alors définir les classes suivantes:

Ac(qi)	ensemble des liaisons aboutissant à qi
Ec(qi)	ensemble des liaisons issues de qi
Aq(qi)	ensemble des variables liées à qi par une liaison y aboutissant.
Eq(qi)	ensemble des variables liées à qi par une liaison issue de qi.

Notations employées par l'algorithme:

Q	variables du graphe
C	liaisons
Qun	variables non classées
Cun	liaisons non classées
qi	variable numéro i
cij	liaison entre la variable qi et la variable qj
PU	variables étant déterminée soit comme paramétre controlable, soit comme input non controlable.
VX	variables étant déterminées soit comme niveau soit comme auxiliaire.
RV	variables étant déterminées soit comme auxiliaire soit comme taux.
F	liaisons classées comme flux
I	liaisons classées comme information
P	input controlable
R	taux
U	input non controlable
V	auxiliaire
X	niveau
Y	output

TABLE 1



Aq(q1) = (q5,q6)
Eq(q1) = (q2)

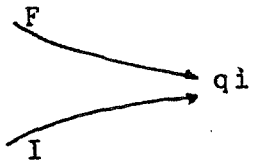
Exemple de diagramme causal avec explicitation des ensembles Aq et Eq

BURNS commence par définir les situations possibles pour les liaisons à un noeud du graphe et introduit une première restriction qui permettra de limiter les recherches de l'algorithme: toutes les liaisons aboutissant à une variable doivent être du même type:

. soit information

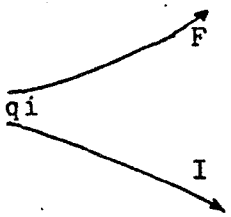
. soit flux

de même toutes les liaisons issues d'une variable doivent être du même type.



est impossible

de même

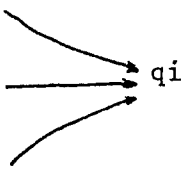


est impossible.

Ce qui signifie en particulier qu'il n'est pas possible d'utiliser la valeur d'un taux comme information pour calculer d'autres variables. Cette restriction est basée sur le fait qu'un taux est en principe une valeur instantanée et ne peut donc être que difficilement mesurable, néanmoins elle supprime une grande part de la souplesse de la dynamique des systèmes qui à ce niveau est très "permissive". De plus cette restriction ne peut se justifier que pour les valeurs de l'instant présent d'un taux les valeurs passées sont très accessibles même pour des phénomènes physiques fugitifs. (FORRESTER emploie une telle information dans INDUSTRIAL DYNAMICS §17.4.3 taux PIF 17.39 ce qui est l'argument décisif ....?)

Après les définitions, BURNS introduit les théorèmes qui permettent de construire son algorithme. En clair ces théorèmes sont les suivants:

1) si aucune liaison n'est issue de qi alors qi est un output

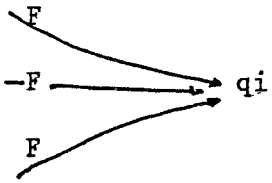


qi output

2) si aucune liaison n'aboutit à qi alors qi est un input

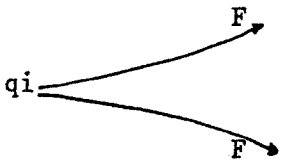
qi input

3) si les liaisons aboutissant à qi sont des flux alors qi est un niveau.



qi niveau

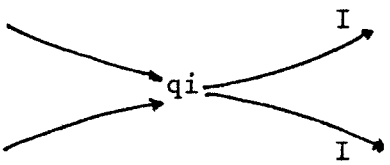
4) si les liaisons issues de qi sont des flux alors qi est un taux.



qi taux

5) si les liaisons issues de qi sont des informations et qu'il existe des liaisons aboutissant à qi alors qi est

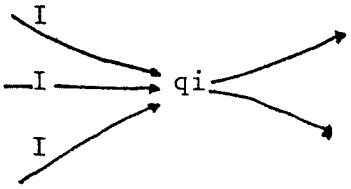
. soit un niveau
. soit une auxiliaire



qi niveau ou auxiliaire

6) si les liaisons aboutissant à qi sont des informations et qu'il existe des liaisons issues de qi alors qi est

. soit un taux
. soit une auxiliaire



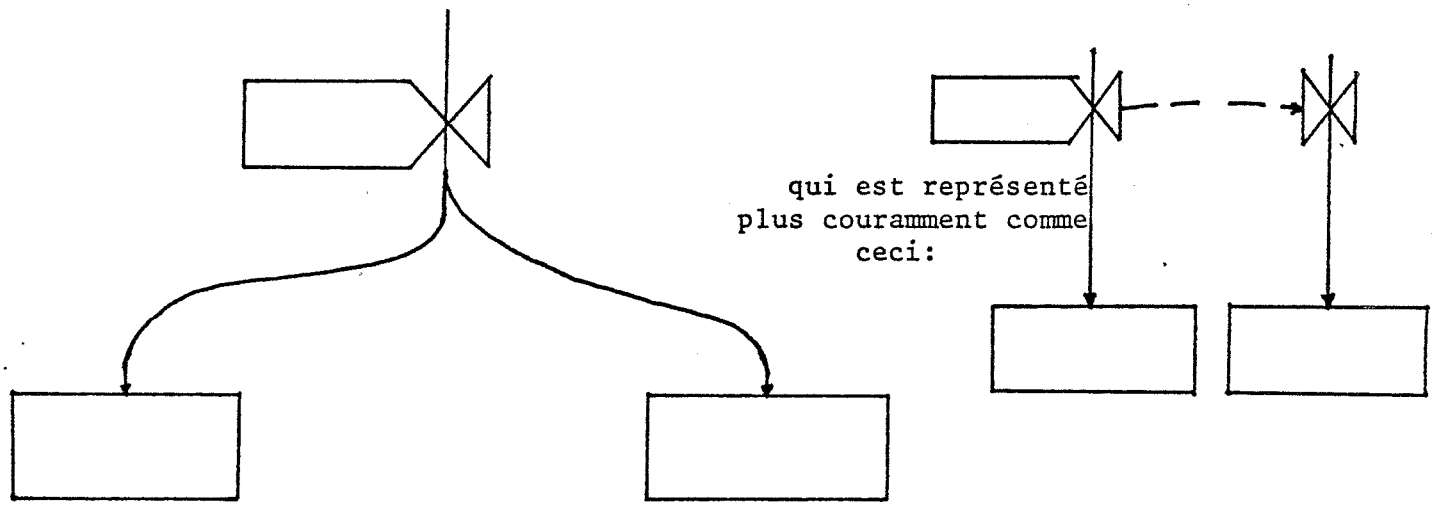
qi taux ou auxiliaire

7) si qi est déterminé comme pouvant être soit un niveau, soit une auxiliaire et si Cij et Cji existent simultanément alors qi est un niveau et qj est un taux.

8) si qi appartient à RV et que le nombre de liaisons issues de qi est supérieur strictement à deux alors qi est une auxiliaire.(un taux ne peut affecter au plus que deux niveaux.)

En tenant compte de ces théorèmes un algorithme peut être établi. Il nécessite, malgré les restrictions introduites, l'aide de l'utilisateur chaque fois qu'une solution ne peut être trouvée.

Une des restrictions est celle imposée par l'axiome 8. Elle signifie qu'un même taux ne peut controler qu'un seul flux ce qui interdit les configurations suivantes:



L'algorithme utilise toutes les remarques précédentes dans une succession d'étapes qui permettent de classer les différentes variables. Toutefois, il arrive que l'intervention de l'utilisateur soit requise quand l'incertitude est trop grande.(étape 8)

L'organigramme est présenté en figure 1 . Les différentes étapes sont les suivantes :

étape 1 initialisation de Qun, Cun, PU, RV, VX, F, I, P, R, U, V, X, Y.

étape 2 recherche des inputs et des outputs et classification des liaisons correspondantes.

a) pour chaque qi appartenant à Qun
   1) si Ec(qi) est vide classer qi dans Y et le retirer de Qun.
   2) si Ac(qi) est vide classer qi dans PU et le retirer de Qun
b) mettre Ac(qi) dans I et retirer ces liaisons de Cun
c) mettre Ec(qi) dans I et retirer ces liaisons de Cun

étape 3 hypothèse de cohérence des liaisons

pour tout cij déjà classé soit dans I soit dans F
   1) classer tous les cik dans la même classe
   2) classer tous les cki dans la même classe

étape 4 trouver des éléments de R et de V grâce à F

pour tout qi de Qun
   si Ac(qi) est inclus dans F mettre qi dans X et le retirer de VX et Qun
   sinon si Ec(qi) est inclus dans F mettre qi dans R et le retirer de RV et Qun.

étape 5 trouver des éléments de V, RV et VX grâce à I

a) pour tout qi de Qun
si Ac(qi) est inclus dans I et Ec(qi) est inclus dans I
alors mettre qi dans V et le retirer de RV, VX et Qun.
sinon si Ac(qi) est inclus dans I mettre qi dans RV
ou si Ec(qi) est inclus dans I mettre qi dans VX
b) si Qun est vide aller à l'étape 9
si Qun n'est pas vide et qu'il n'y a pas de nouveaux éléments dans RV ou VX alors aller à l'étape 8

étape 6 recherche de sous-modèles contenant des éléments de VX

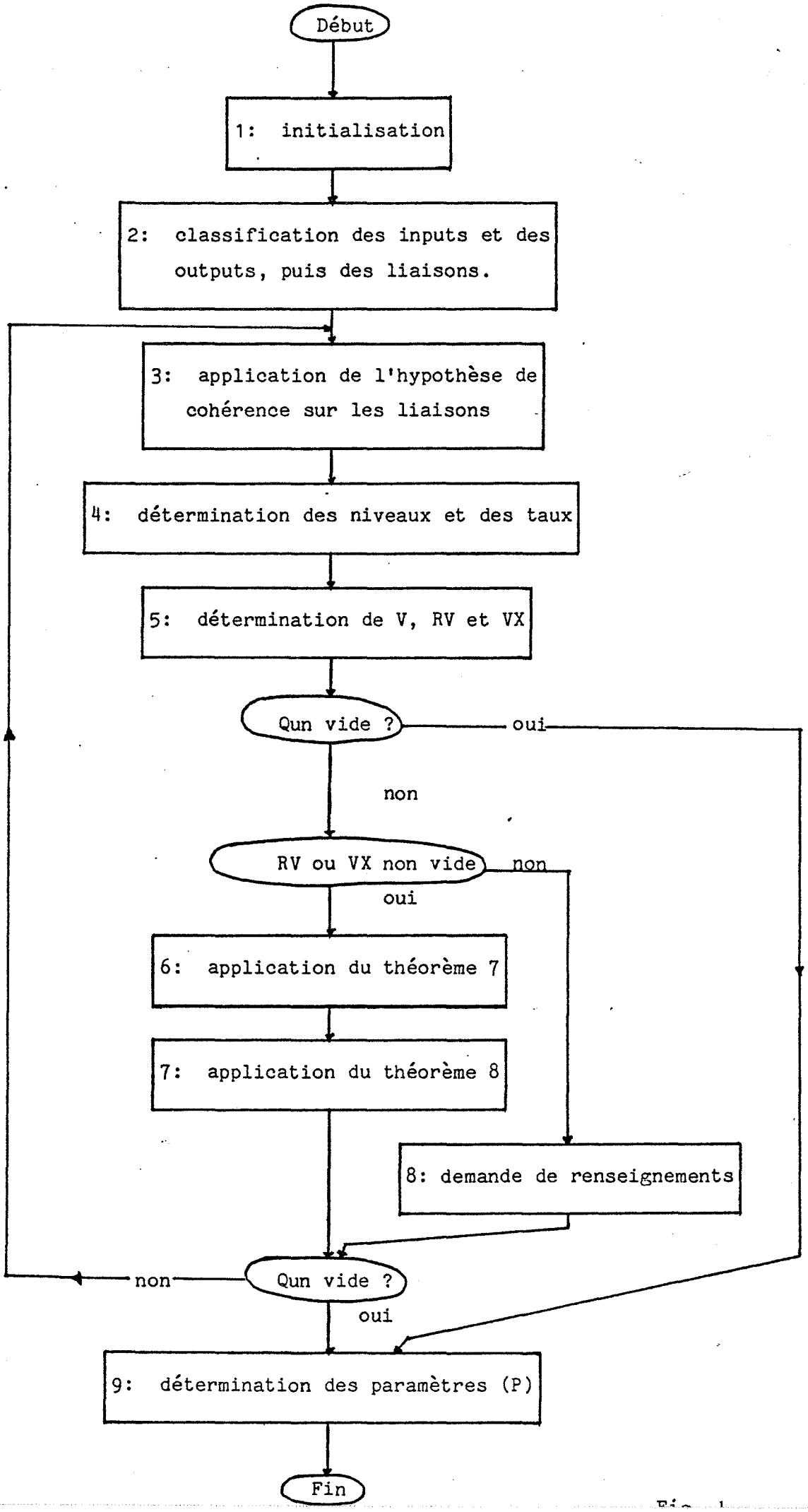
pour tout qi de VX si cij et cji existent
a) mettre qi dans X et le retirer de VX et Qun
b) mettre qj dans R et le retirer de RV et Qun
c) mettre cij dans F et le retirer de Cun

étape 7 étude du cardinal de Ec(qi) des qi appartenant à RV

pour tout qi de RV si Card(Ec(qi)) est strictement supérieur à deux alors
a) mettre qi dans V et le retirer de RV et Qun
b) mettre Ec(qi) dans I et les retirer de Cun
Si Qun est vide aller à l'étape 9, sinon aller à 3

étape 8 demande de renseignements à l'utilisateur

choisir un élément de Qun et spécifier sa classe
si qi appartient à X alors
a)mettre qi dans X et le retirer de VX et Qun
b)mettre les éléments de Ec(qi) non classés dans I
c)mettre les éléments de Ac(qi) non classés dans F
si qi est un élément de R alors
a)mettre qi dans R et le retirer de RV et Qun
b)mettre les éléments de Ec(qi) non classés dans F
c)mettre les éléments de Ac(qi) non classés dans I


Début
1: initialisation
2: classification des inputs et des outputs, puis des liaisons.
3: application de l'hypothèse de cohérence sur les liaisons
4: détermination des niveaux et des taux
5: détermination de V, RV et VX
Qun vide ?
oui
non
RV ou VX non vide
non
oui
6: application du théorème 7
7: application du théorème 8
8: demande de renseignements
Qun vide ?
non
oui
9: détermination des paramètres (P)
Fin

si qi est un élément de V alors

a)mettre qi dans V et le retirer de RV, VX et Qun

b)mettre les éléments de Ac(qi) et Ec(qi) non classés dans I

Si Qun est vide aller à l'étape 9 sinon à l'étape 3.

étape 9 répartition éventuelle des éléments de PU dans les deux classes P et U suivant les indications de l'utilisateur.

Fin de l'algorithme.

Nous avons programmé cet algorithme et essayé sur différents diagrammes, dont certains pris dans la littérature. Les essais et les conclusions sont présentés en annexe.

. Méthode employée dans Dynamine

BURNS impose deux restrictions pour pouvoir utiliser son algorithme: les taux ne peuvent pas fournir d'information et ne peuvent affecter que deux niveaux au plus (un en sortie et un en entrée). Nous avons opté pour moins de restrictions au détriment de l'automatisation (la différence est pourtant minime). Nous avons utilisé les remarques suivantes:

- à un niveau ne peuvent aboutir que des liaisons issues d'un taux et ces liaisons ne peuvent être que des flux. Donc si les niveaux sont connus, les taux le sont (à part le cas où un taux est isolé et n'influe sur aucun niveau.)

- toutes les autres variables sont des auxiliaires qui peuvent être classées en trois groupes:
  . output
  . input
  . autre

De ces remarques nous avons abouti à la procédure utilisé dans Dynamine:

- dans un premier temps l'utilisateur désigne les niveaux,qui sont en général les quantités les plus facilement identifiables, la difficulté étant de faire la différence entre les taux et les auxiliaires dans un graphe fortement connecté.

- muni de ces renseignements, le programme détermine quels sont les taux grâce à la première remarque et en déduit les flux.

- toutes les autres liaisons ne peuvent être que des liaisons d'information donc I est déterminé.

- les variables non encore classées sont des auxiliaires que l'on peut diviser en input, output et autres grâce à une procédure semblable à celle de BURNS.

Cette méthode demande autant d'interventions qu'il y aura de niveaux dans le modèle. La comparaison avec la méthode de BURNS en termes de nombre d'interventions n'est pas significative. Par contre, au niveau de l'emploi, on peut dire que la méthode de BURNS peut créer des surprises quand le choix se présente devant l'utilisateur et que des difficultés surgissent suite aux choix précédents ou à une réponse inattendue du système. Notre méthode donne une plus large part au contrôle de l'utilisateur, car c'est lui en fait qui choisit sa structure par le choix des niveaux. Si il n'est pas satisfait il peut modifier complètement le choix des niveaux et obtenir une structure très différente.

On le voit clairement, le choix entre les deux méthodes est une question de goût plus que de raison.

b) Ecriture des équations

Après avoir construit le diagramme causal et déterminé le type des variables, il ne reste plus qu'à écrire les équations du modèle, c'est à dire quantifier les relations définies par l'étape précédente. Pour cela un éditeur de texte spécialisé a été créé et intégré à Dynamine.

Avant de décrire cet éditeur rappelons les informations déjà connues du système à ce stade.

- les noms des variables
- les relations
- le type des variables

Ces informations sont suffisantes pour générer les équations de niveaux qui sont du type :

NIV(K) = NIV(J) + DT * (somme des flux en relation avec NIV affectés du signe de la relation)

Ces équations seront donc générées automatiquement, l'utilisateur n'ayant alors plus qu'à préciser les valeurs initiales des niveaux. Dans un modèle moyen on peut estimer que le pourcentage d'équations ainsi générées varie entre 20 % et 25 %, ce qui élimine une bonne part de travail fastidieux et dépourvu de toute créativité, ce qui est le but d'un système de conception assistée.

Les informations accessibles permettent de présenter l'éditeur sous la forme d'un menu affiché qui contient (figure 2)

- les opérateurs habituels +,-,*,/ et **
- les fonctions usuelles FORTRAN
- les fonctions usuelles de la dynamique des systèmes
- les noms des variables en relation avec la variable dont on écrit l'équation.
- les fonctions éditeur de texte

+	-	*	/	**	(J)	(K)	DT	(	)	,	=			
DELAI	ABS	FLOAT	AINT	AMOD	AMAX1	AMIN1	SIGN	DIM	EXP	ALOG	ALOG10	SQRT	SIN	COS
TANH	ATAN	RAN	P3SUP4	PA3ZER	TABLE	TABLEH	ESCALI	SAUT	ALEAT	ALEASM	PULSAT	LISINF		
INSERT	SUPPRE	INIVAR	INIPAR	FIN										
A	B													

Menu affiché par l'éditeur de texte spécialisé

Fig. 2

Ce qui fait que le clavier sera utilisé le moins possible. Néanmoins pour laisser une certaine souplesse d'utilisation, les équations peuvent être écrites de façon normale au clavier. On peut d'autre part ajouter en cours d'édition des variables non initialement prévues dans le diagramme ou ne pas utiliser toutes les variables indiquées en relation. Dans ce cas la cohérence entre les équations et le diagramme n'est plus totalement assurée.

Quand l'équation a été écrite entièrement, un contrôle de syntaxe est effectué, ce qui permet la rectification immédiate des erreurs détectées sans attendre une compilation éventuelle du modèle avant l'exécution.

c) Construction du diagramme des flux - Editeur graphique

Si l'utilisateur a déjà défini le diagramme causal de son modèle et déterminé le type des variables, il peut pour des raisons de clarté préférer éditer le diagramme des flux. Ce diagramme est porteur d'informations aussi complètes que le diagramme causal, avec l'avantage que les variables sont classées par type. Il est d'autre part plus imagé et donc plus parlant pour des personnes qui ne connaissent pas la dynamique des systèmes et permet donc un meilleur dialogue entre le modélisateur et ses interlocuteurs.

Pour construire un tel diagramme, il est nécessaire de disposer d'un éditeur graphique qui permette de placer les symboles de la dynamique des systèmes et de les enchainer. Cet éditeur s'appliquant à des structures en réseau et non à des chaines de symboles, il devra être d'un type un peu particulier, bien qu'il doive disposer des fonctions de base:

- insertion de symboles
- suppression
- modification
- désignation ou recherche d'un élément
- affichage.

Le travail de l'utilisateur devant être facilité au maximum, nous avons introduit au niveau de l'éditeur un contrôle "syntaxique" qui permet de signaler les erreurs les plus grossières et de les rectifier immédiatement. Il faut donc définir une sorte de grammaire graphique rejetant les séquences non valides.

D'autre part, cet éditeur étant graphique, il devra comporter les fonctions élémentaires graphiques permettant de manipuler des schémas sur un écran :

- fenêtrage
- zoom
- déplacement de la fenêtre

Ces fonctions sont rendues nécessaires par le fait qu'un écran est limité en surface et qu'en général le plan utilisateur (plan virtuel sur lequel travaille le modélisateur) est nettement plus vaste.

La conjonction de ces divers points, plus de quelques remarques sur la nature des réseaux à dessiner, ont abouti à l'éditeur graphique de Dynamine.

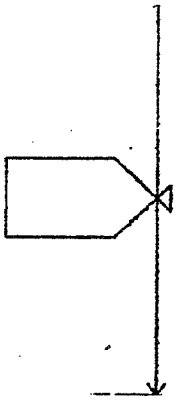
L'écran est divisé en deux zônes distinctes:

- une "clôture" sur laquelle est projetée la "fenêtre" utilisateur.
- le reste de l'écran où se placeront les "menus" présentant les possibilités d'action et les messages du dialogue.

Pour construire le diagramme, l'utilisateur dispose de symboles prédéfinis qui sont les symboles de bases de la dynamique des systèmes. Les noms des variables correspondantes à ces symboles sont édités à côté du symbole (si l'échelle d'affichage le permet) Les symboles sont visibles sur la figure3 .

Puits

Vanne



Niveau

Auxiliaire

constante

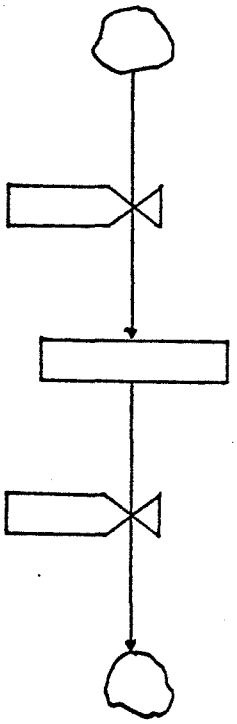
liaison d'information ←----------

Symboles utilisés par l'éditeur graphique.

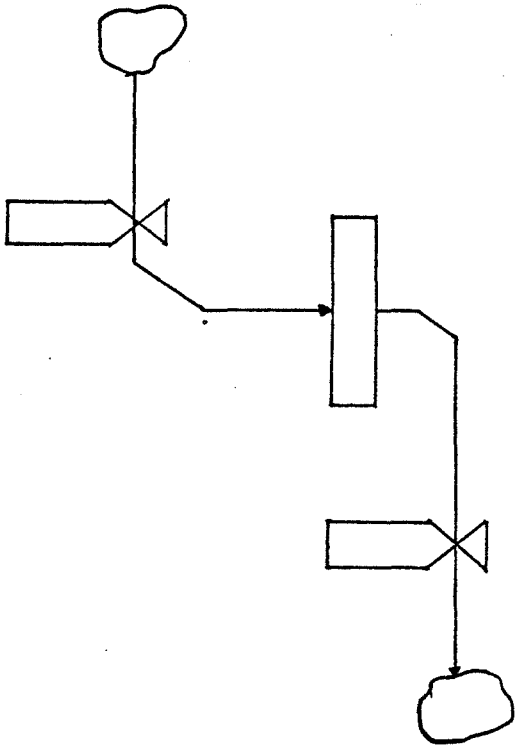
Fig. 3

Un certain nombre de remarques permet une construction plus aisée et plus rapide par le placement automatique des symboles en utilisant une notion de contexte.

Un réseau dans un diagramme de flux a rarement une forme "torturée", souvent il se compose d'une suite de symboles en ligne droite, mis à part les bifurcations et les jonctions éventuelles. L'emplacement des différents symboles est donc connu quand le point de départ et la direction générale du réseau sont connus.



est la situation
la plus courante
et non pas :

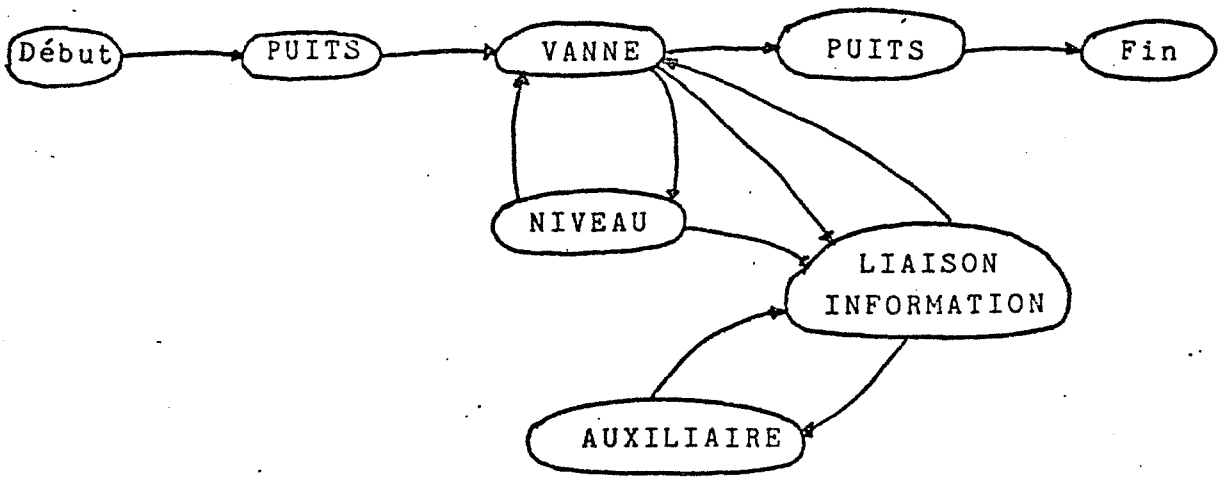


Donc pour construire un réseau, l'utilisateur n'aura à indiquer que l'emplacement du premier puits et la direction générale : horizontale ou verticale. Les différents symboles se placent automatiquement les uns à la suite des autres dès que l'utilisateur a désigné le suivant. Le dernier symbole placé definit le contexte par défaut qui sera utilisé pour le contrôle syntaxique.

L'utilisateur peut néanmoins modifier l'emplacement du symbole à placer. S'il veut modifier la direction seulement, il lui suffira de l'indiquer et le réseau fera un coude. Si par contre il veut engendrer une branche nouvelle, il devra modifier le contexte

et indiquer pour cela le nouveau en désignant grâce au réticule un symbole déjà affiché. Ainsi avec peu de manipulations le réseau est construit.

Le contexte correspond donc soit au dernier symbole placé, soit à un symbole désigné par l'utilisateur. Le contrôle de syntaxe se fait en comparant la suite "contexte, nouveau symbole" avec les suites autorisées par la grammaire.(voir automate fig. 4)



Automate avec liaison d'information parmi les états

Fig. 4

Les liaisons d'informations peuvent être définies à n'importe quel moment en désignant le symbole origine, les points par lesquels on veut faire passer cette liaison, le symbole arrivée. Un contrôle est effectué pour vérifier si, suivant la syntaxe autorisée, les deux symboles peuvent être en liaison et dans ce sens.

La grammaire permettant le contrôle est simple. Rappelons en les grandes lignes:

- après un puits, il ne peut y avoir qu'une vanne

- après une vanne, puits ou niveau

- après un niveau, vanne

- les auxiliaires sont reliées au reste du réseau uniquement par des liaisons information.

- une liaison information ne peut pas aboutir à un niveau ou un puits

- une liaison information ne peut pas être issue d'un puits

L'utilisateur construit son diagramme tout en indiquant le nom des variables. Cette étape effectuée, nous avons les informations suivantes :

- noms des variables
- type des variables
- relations
- sens des relations

donc comme pour le diagramme causal, les équations de niveaux peuvent être générées automatiquement et on peut utiliser l'éditeur de texte spécialisé avec l'ensemble de ses possibilités.

C. Le logiciel graphique

La nécessité d'un logiciel graphique est apparue rapidement, car il ne suffisait pas d'afficher quelques segments de droite mais des figures structurées, composées de symboles graphiques divers et de texte. La non adéquation des logiciels existant poussant, il a été nécessaire de développer ce logiciel qui est devenu un véritable éditeur graphique pour des scènes planes.

Ce logiciel doit répondre à divers impératifs:

- stockage des données sous une forme structurée et condensée
- affichage des figures
- modifications nombreuses
- interaction avec l'utilisateur

De plus, voulant réaliser un système transportable dans la mesure du possible, il a été obligatoire de prévoir une indépendance vis à vis du matériel pour la plus grande partie du logiciel.

Nous nous sommes inspirés de plusieurs logiciels existant (GRIGRI, LOGIGRAPH etc...) pour différentes notions de base, mais la structure globale est originale pour une large part.

a) les notions de base

Le but est d'afficher sur un écran graphique des images structurées, fabriquées à partir de segments de droite et de texte. Ceci amène deux types de notions :

- les notions liées à la structure de l'image
- les notions liées au support physique de l'image ou de la portion d'image affichée.

Pour la structure de l'image, ce qui vient à l'esprit immédiatement est de constituer des "programmes graphiques" composés d'images constituées de sous-images, le dernier niveau de la hiérarchie étant le segment de droite et le caractère. Donc une figure sera en fait une "image" faisant appel à d'autres images qui elles-mêmes font appel à des images et ainsi de suite récursivement jusqu'au niveau de l'instruction élémentaire que constituent les segments de droite et les caractères. La récursivité dans les appels aux images n'est pas interdite et peut même présenter des avantages pour étudier différentes structures.

Des paramètres sont nécessaires pour préciser la structure des figures:

- place des images dans le plan
- orientation de ces images
- taille
- éventuellement couleur ou niveau de gris et intensité lumineuse.
- caractéristiques du trait . grosseur
  . tireté, pointillé etc...
- taille des caractères

Des attributs logiques peuvent aussi être nécessaires:

- désignabilité
- visible ou non visible

- possibilité de suppression
- liens avec d'autres données

etc...

Tous ces paramètres sont précisés après l'instruction "image" en plus de la désignation de l'image appelée. Une figure quelconque sera le résultat d'un programme du type suivant:

image 1: image 2 (attributs et paramètres)
segments de droite (coordonnées)
image 3 (attributs et paramètres)
image 4 (attributs et paramètres)

image 2: image 3 (attributs et paramètres)
segments de droite (coordonnées)
image 3 (attributs et paramètres)

.
.
.
.

Nous avons donc un langage graphique comprenant les instructions suivantes:

- Suite de segments de droite (coordonnées)
- Texte (taille place texte)
- Appel à image (nom image, place, taille, rotation)

Deux emplois d'un tel langage sont possibles:

l'affichage d'une structure en évolution (l'affichage s'effectue au fur et à mesure de la construction : DYNAMINE)

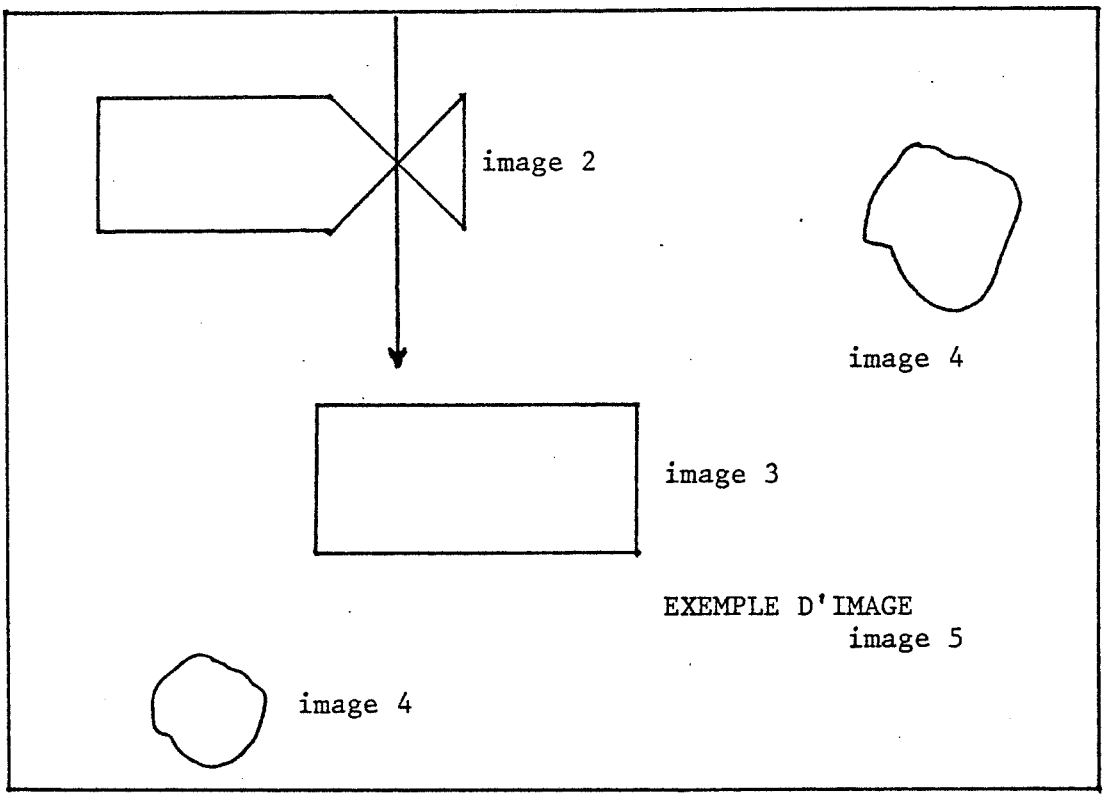
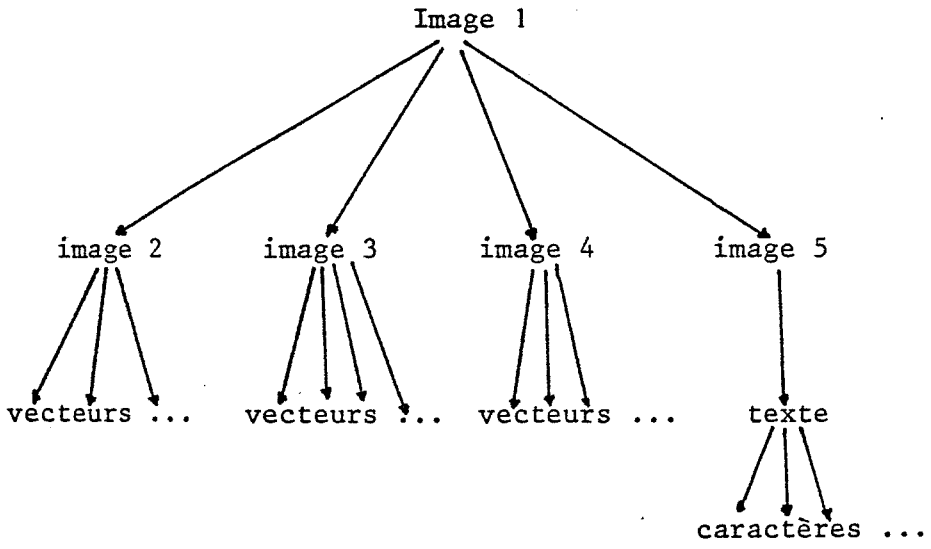


Image 1

Structure de l'arbre d'appel :



(l'image 4 est appelée 2 fois)

Exemple de scène graphique et d'arbre d'appel correspondant.

Fig. 5

l'affichage d'une structure figée (synoptique pour du contrôle de processus industriel par exemple)

avec des gradations entre ces deux extrèmes.

Le premier type d'utilisation oblige à exécuter le programme graphique correspondant en interprété du fait des modifications nombreuses. Le deuxième peut justifier une "compilation" qui permettra un affichage rapide de la structure au prix d'une perte de place, le "code compilé" étant plus volumineux.

Les modifications étant fréquentes dans Dynamine, nous avons opté pour une interprétation de la "liste d'affichage", le temps d'affichage, de 10 à 20 secondes au maximum, n'étant jamais prohibitif, même pour une structure fouillée.

Il est nécessaire d'afficher les figures sur un support et donc il faut préciser d'autres notions. Le support dont on dispose est fini, il faudra donc se limiter à une portion finie du plan sur lequel est dessinée la figure, cette portion du plan virtuel utilisateur est appelée "fenêtre" et dans notre cas sera toujours un rectangle déterminé par les coordonnées des coins bas gauche et haut droit. De même, il faudra projeter ce morceau de plan utilisateur sur l'écran. Pour définir l'emplacement de cette projection nous utiliserons la notion de "clôture" qui comme la fenêtre sera un rectangle. La fonction d'affichage consistera donc à projeter la "fenêtre" du plan utilisateur sur la portion d'écran définie par la "clôture.

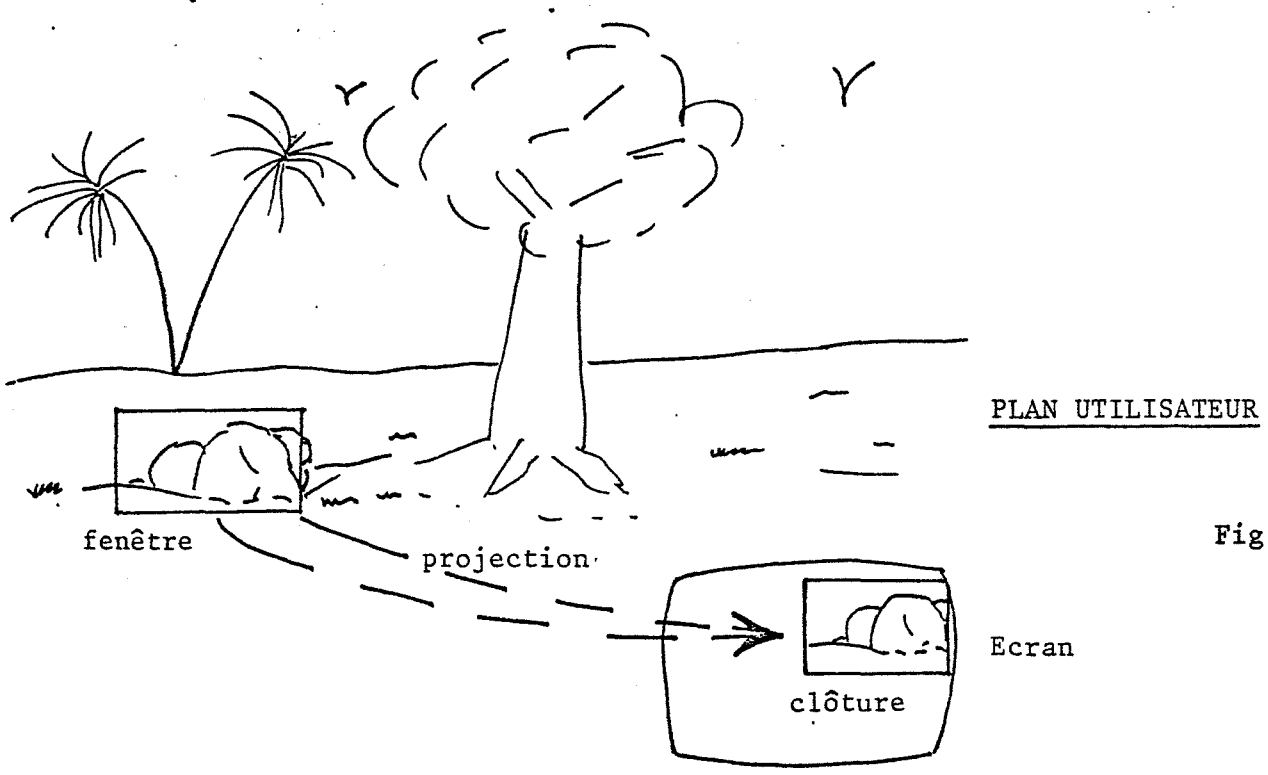


Fig. 6

Ces différentes notions permettent à peu près toutes les opérations graphiques voulues pour l'affichage d'une scène quelconque et sa structuration. Il ne manque que le niveau interaction, pour cela nous avons utilisé les techniques classiques:

- menu
- réticule pour la désignation d'un point ou d'une image
- le clavier

Structure de l'éditeur graphique

La structure de cet éditeur graphique répond à la conception que nous avons du travail effectué pour construire une figure. Nous supposons qu'il y a deux phases pouvant être simultanées mais différentes logiquement:

- la construction de la figure, c'est à dire pour reprendre le parallèle avec un programme informatique, l'édition et le stockage du programme graphique.

- l'affichage, c'est à dire l'exécution.

Pour une meilleure interaction, il est possible d'afficher en même temps que l'on construit la figure.

Le logiciel graphique sera constitué de deux grandes parties, l'une consacrée à l'élaboration des figures et qui sera donc l'éditeur proprement dit, et la seconde, très indépendante, destinée à l'affichage des scènes existantes à la demande de l'utilisateur ou automatiquement durant la construction d'une figure si cette option a été demandée.

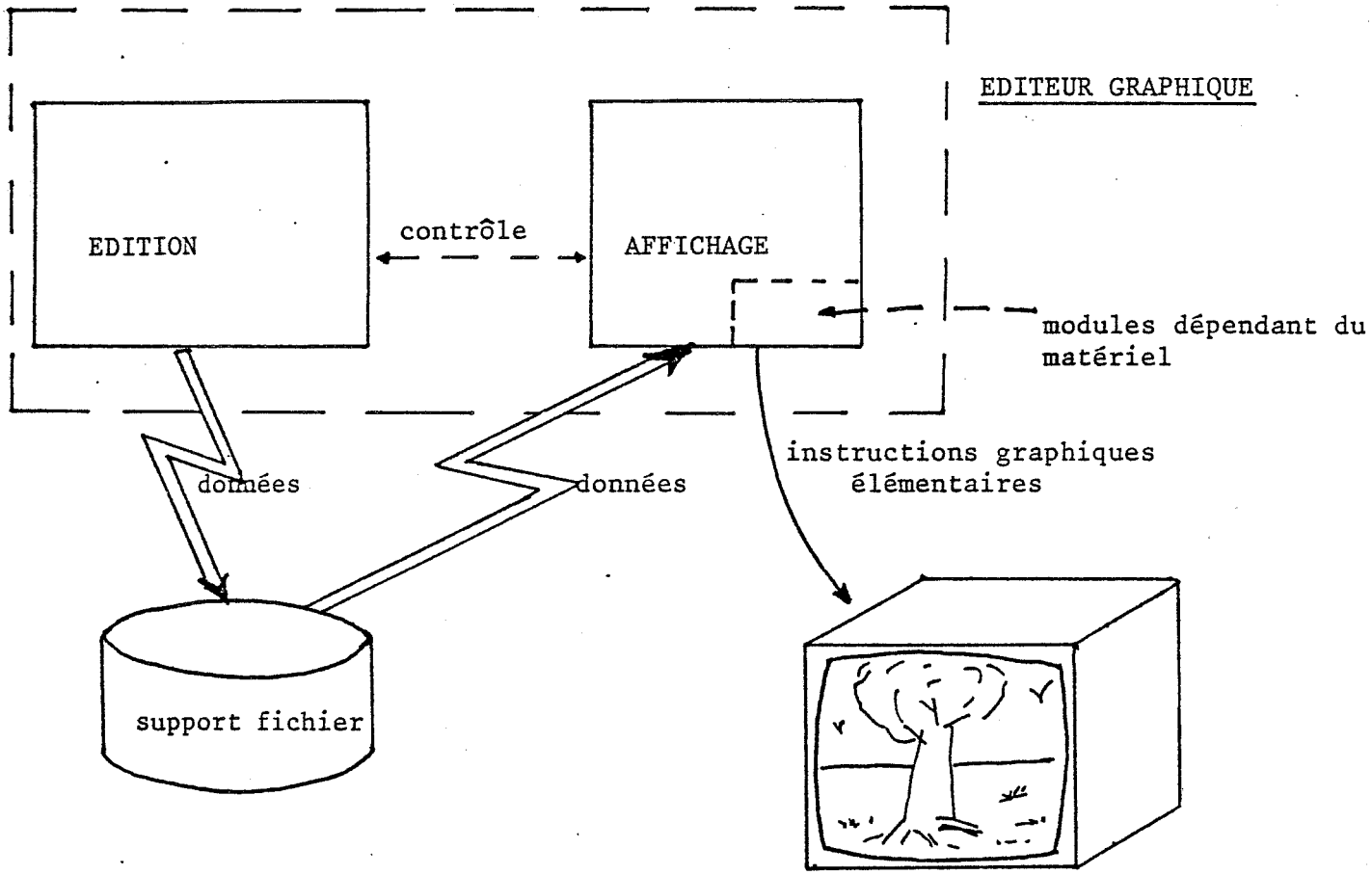


Fig. 7

Nous voyons sur ce schèma que la partie édition est totalement indépendante du matériel de visualisation, seule la partie affichage peut en être dépendante. Pour que le matériel impose le moins de restriction possible, nous avons adopté le découpage suivant:

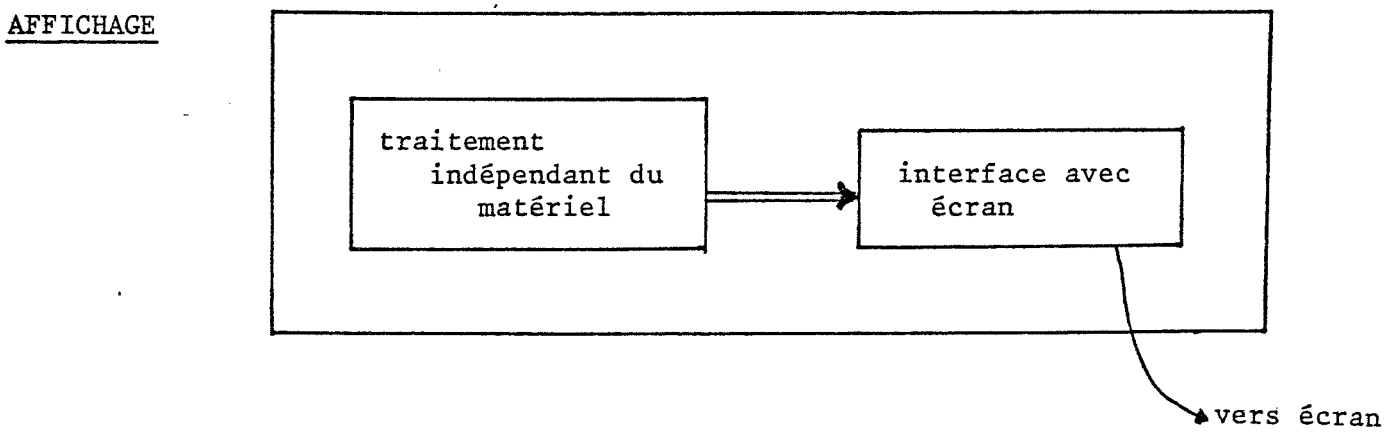


Fig. 8

Les modules d'interface ne correspondent qu'à ce qui est strictement dépendant du terminal: codage des instructions graphiques élémentaires.

Le traitement indépendant correspond en fait à la transformation de la structure des données en une liste d'affichage en coordonnées utilisateur, puis au fenêtrage et transformation en coordonnées écran (en supposant un écran virtuel de taille fixée une fois pour toutes, le module dépendant de l'écran faisant une dernière transformation d'échelle).

D Exécution et exploitation

Quand le modèle a été construit et les équations écrites, il ne reste plus qu'à l'exploiter pourvu qu'il n'y ait pas d'erreurs de conception ou d'écriture. Ces phases de mise au point et d'exploitation font intervenir les modules d'exécution et d'exploitation des résultats. Ces deux parties du système doivent être d'un emploi le plus pratique possible et permettre le recul nécessaire vis à vis de la technique pour que le modélisateur ne disperse pas son attention vers des aspects mineurs.

L'exécution se déroulera donc de la façon suivante:

- tri des équations
- initialisation, si des variables ne sont pas initialisées explicitement, une initialisation automatique est effectuée pour celles-ci.
- demande de la liste des variables dont le comportement doit être suivi et donc enregistré.
- demande des paramètres DT et DUREE
- exécution proprement dite.

Le tri des équations s'effectue entièrement sans intervention de l'utilisateur et n'appelle aucun commentaire, si ce n'est que c'est un moyen de détecter les équations croisées du type:

```
F1(K) = f1(F2(K),....)
F2(K) = f2(F1(K),....)
```

qui ne sont pas permises.

L'initialisation est un aspect important de l'exécution. Il y a en fait deux traitements distincts: celui des variables initialisées explicitement et celui des variables non initialisées. On commencera par les premières et grâce à leur valeurs on pourra initialiser les suivantes en utilisant une hypothèse de stationnarité à l'initialisation.

Pour vérifier les résultats de cette initialisation, l'utilisateur a accès aux valeurs des différentes variables et peut s'il le désire les modifier, ce qui permet de faire facilement les tests de sensibilité du modèle face à des variations des conditions initiales ou de certains paramètres.

L'exécution proprement dite ne demande aucune action de la part de l'utilisateur, une fois le processus déclenché, mais les erreurs et différents incidents pouvant intervenir lui sont signalés de façon explicite. Les messages d'erreur seront ainsi fournis avec l'indication de l'équation en cause et de l'instant d'occurence. De plus l'utilisateur a accès aux valeurs décrivant la trajectoire du système dans l'espace à n dimensions des n variables considérées comme importantes. Mais ceci n'est pas suffisant, pour permettre une meilleure mise au point,il faut disposer de points d'arrêt programmables sur des situations, ainsi l'utilisateur pourra tester le comportement du modèle dans ses moindres détails et palier aux déficiences éventuelles dans l'écriture des équations.

Les résultats ont en général une certaine importance pour le modélisateur, l'accès à ces résultats est donc un point capital. Ils devront d'abord être enregistrés sous une forme qui permette toutes les manipulations et ceci indépendamment du système d'aide à la modélisation: ils seront donc enregistrés sur un fichier comportant les renseignements nécessaires à leur utilisation:

- nom des variables
- DT et DUREE
- valeurs enregistrées

Ce qui permettra de réutiliser ce fichier à tout moment, pour tout calcul éventuel.

Le système pour sa part permet de présenter les résultats sous deux formes, tableaux de valeurs et diagrammes des trajectoires, ce qui donne une bonne idée du comportement du modèle indépendamment de tout test statistique ou autres.

Les tableaux de valeurs permettent de sortir les valeurs numériques d'un intervalle de temps donné suivant un pas dt donné. Ils ne peuvent que difficilement servir à visualiser le comportement du système mais permettent par contre une vérification précise de différents détails. On retrouve là une nouvelle fois la dualité information compréhension, une grande quantité d'information sous une forme brute nuisant à une bonne compréhension globale du phénomène étudié.

L'accent a été mis sur la présentation des courbes. Ce traitement est entiérement interactif, l'utilisateur pouvant afficher simultanément sur l'écran jusqu'à quatre diagrammes différents, de tailles variables et aux emplacements choisis grâce au réticule. Ces diagrammes peuvent comporter dix courbes différentes soit à la même échelle (dans la mesure où cela signifie quelque chose) soit chacune occupant tout le champ en ordonnée , ce qui permet de comparer les variables et leur

évolution. Ces opérations peuvent être renouvelées autant de fois que l'utilisateur le désire et il peut ainsi choisir la meilleur présentation des différents résultats avec les associations les plus pertinentes.

Ce traitement des résultats n'est qu'un préambule à différents tests que l'utilisateur devra effectuer. Il permet de se rendre compte de la correction du comportement du modèle du moins par rapport à ce que l'utilisateur attendait et de rechercher la cause des divergences éventuelles. Les tests ultérieurs devront être écrits par l'utilisateur, ceux-ci étant trop variés et trop spécifiques du modèle pour être intégrés au système.

Les phases d'exploitation seront les suivantes:

- validation du modèle

  c'est une phase qui vérifie que le modèle a un comportement semblable (terme à définir) à celui du système étudié, pour les situations connues.

- tests de sensibilité aux variations des conditions initiales et des paramètres importants.

- exploitation du modèle dans un but de prédiction en général, ou pour étudier les comportements "optimaux" du système en faisant varier les paramètres.

Notons par ailleurs, que différents programmes de traitements graphiques des données sont utilisables:

- un programme permettant d'afficher des diagrammes représentant l'évolution d'une variable en fonction d'une autre et non plus du temps, ce qui permet d'obtenir les diagrammes de phase.
- un programme permettant l'étude des surfaces de réponse aux variation d'un paramètre (voir exemple en annexe)

Annexe: applications de l'algorithme de BURNS

Le premier modèle étudie le développement d'une zône résidentielle. Son auteur, M. GOODMAN, donne le diagramme causal présenté en figure . BURNS a présenté son algorithme avec cet exemple et nous l'avons repris pour tester notre programme.

Les résultats sont les suivants (semblables à ceux de BURNS)

- le programme demande une première intervention. Les variables non classées sont: ATTR HAV LFO LAV.
  Nous avons désigné ATTR comme auxiliaire.

- une deuxième intervention est nécessaire une ambiguité subsistant sur LFO et LAV.
  Nous avons désigné LFO comme auxiliaire.

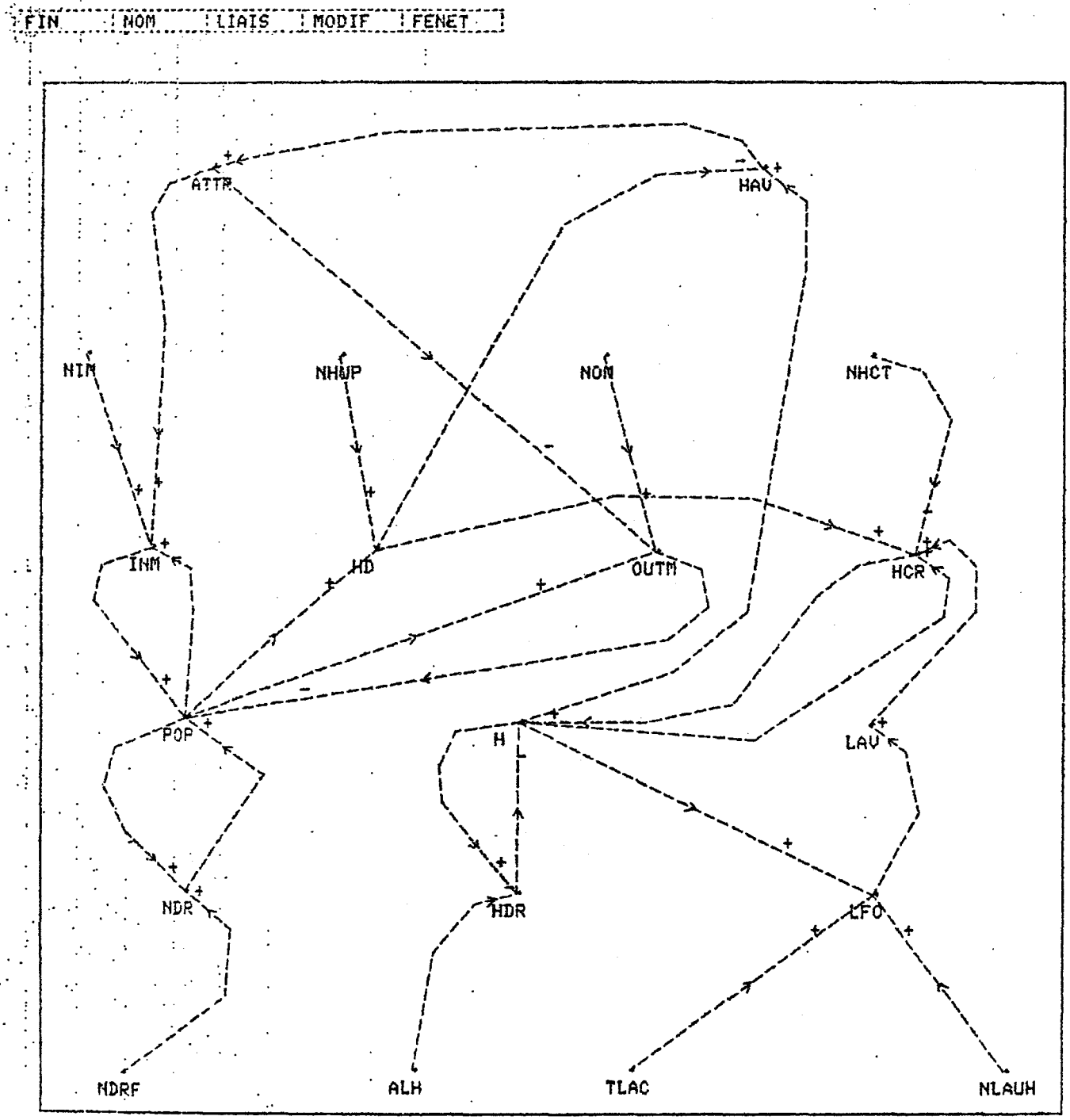
Le classement définitif est celui de la figure . Comme on le voit pour un modèle comportant deux niveaux il a fallu deux interventions, ce qui correspond à notre algorithme. L'algorithme de BURNS donne de meilleurs résultats si nous désignons LAV puis LAV et ATTR comme niveaux. Dans ces deux cas avec deux interventions nous obtenons respectivement trois puis quatre niveaux.

Nous pouvons conclure pour l'instant que l'algorithme de BURNS opère de manière plus automatique que notre méthode, ce qui est évident. Mais les problèmes se posent quand on opère sur des modèles moins bien adaptés à ce traitement (voir l'exemple du modèle de l'université).

Les essais suivants portent sur le modèle du plateau de KAIBAB (GOODMAN), le modèle de la fièvre jaune (K. KALGRAF) puis le modèle des flux d'étudiant dans une université. Dans les deux premiers cas les essais sont concluants, surtout pour le modèle de

la fièvre jaune qui ne nécessite que trois interventions qui déterminent la structure complète de 8 niveaux, 10 taux et 3 auxiliaires. Par contre pour le modèle de l'université un problème se pose. Dans ce cas aucune intervention n'est requise, mais le fait que la variable NINCR1 soit un INPUT détermine toute la structure de sorte que nous obtenons un résultat inverse de ce nous attendions. Les variables que nous considérions comme des niveaux sont déterminées être des taux et vis-versa. On obtient donc une structure totalement inattendue et qui n'a que peu de rapport avec la logique du modèle. Dans ce cas on voit que la procédure automatique est génante.

En conclusion nous pouvons dire que pour des modèles dont la structure est bien conforme aux axiomes de BURNS, son algorithme est parfait, mais que dans les autres cas il présente des inconvénients génants qui empêchent de l'utiliser. (précisons que pour le modèle de la fièvre jaune, cas où l'algorithme donne toute sa puissance, il a fallu truquer un peu en ajoutant une relation entre PDM et BM qui n'existait pas dans le modèle original, BM étant alors un INPUT)



Residential community model

(M. R. GOODMAN)

Fig. 1

```
** RESULTAT TRANSFORMATION **
   ***********************

 NIVEAU

POP      H

 TAUX

NDR      INM      OUTM     HCR      HDR

 AUXILIAIRES

HD       ATTR     HAV      LFO      LAV

 INPUT

NIM      TLAC     ALH      NOM      NDRF     NHUP     NHCT     NLAUH
```

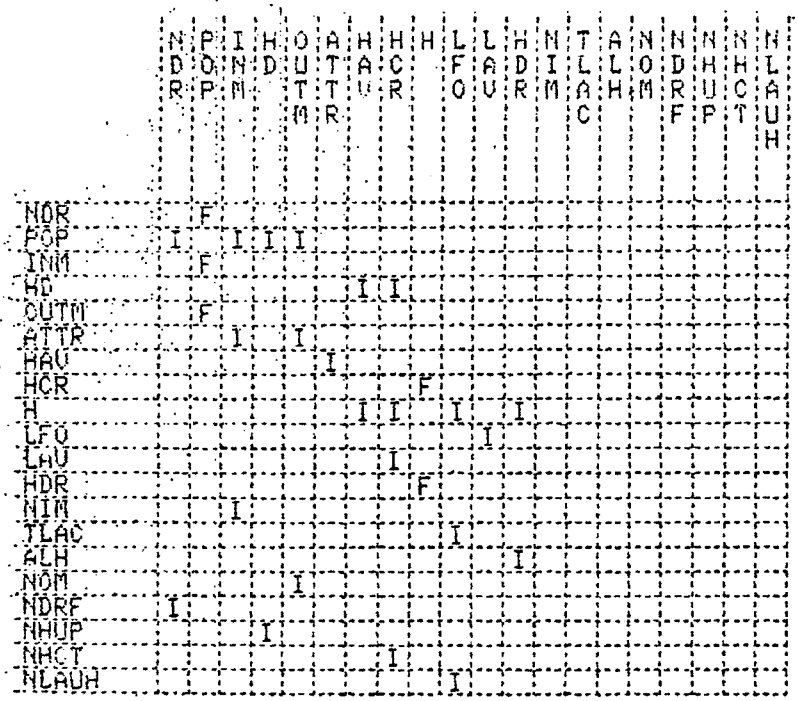


Fig. 2

```
        ** RESULTAT TRANSFORMATION **
           **********************

 NIVEAU

POP       H         LAV

 TAUX

NDR       INM       OUTM      HCR       LFO       HDR

 AUXILIAIRES

HD        ATTR      HAV

 INPUT

NIM       TLAC      ALH       NOM       NDRF      NHUP      NHCT      NLAUH
```

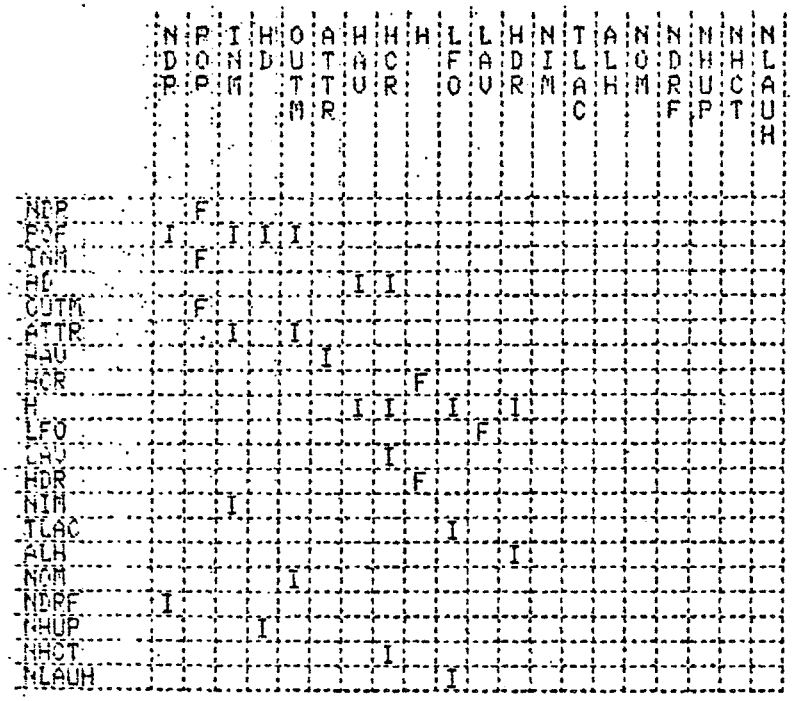


Fig. 3

** RESULTAT TRANSFORMATION **
****

NIVEAU

POP ATTR H LAU

TAUX

NDR INM OUTM HAU HCR LFO HDR

AUXILIAIRES

HD

INPUT

NIM TLAC ALH NOM NDRF NHUP NHCT NLAUH

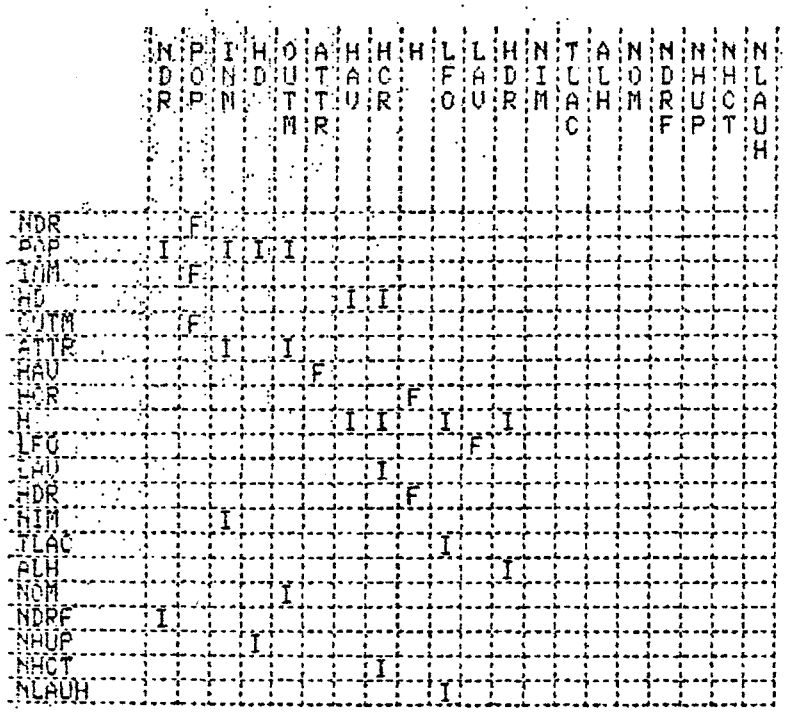
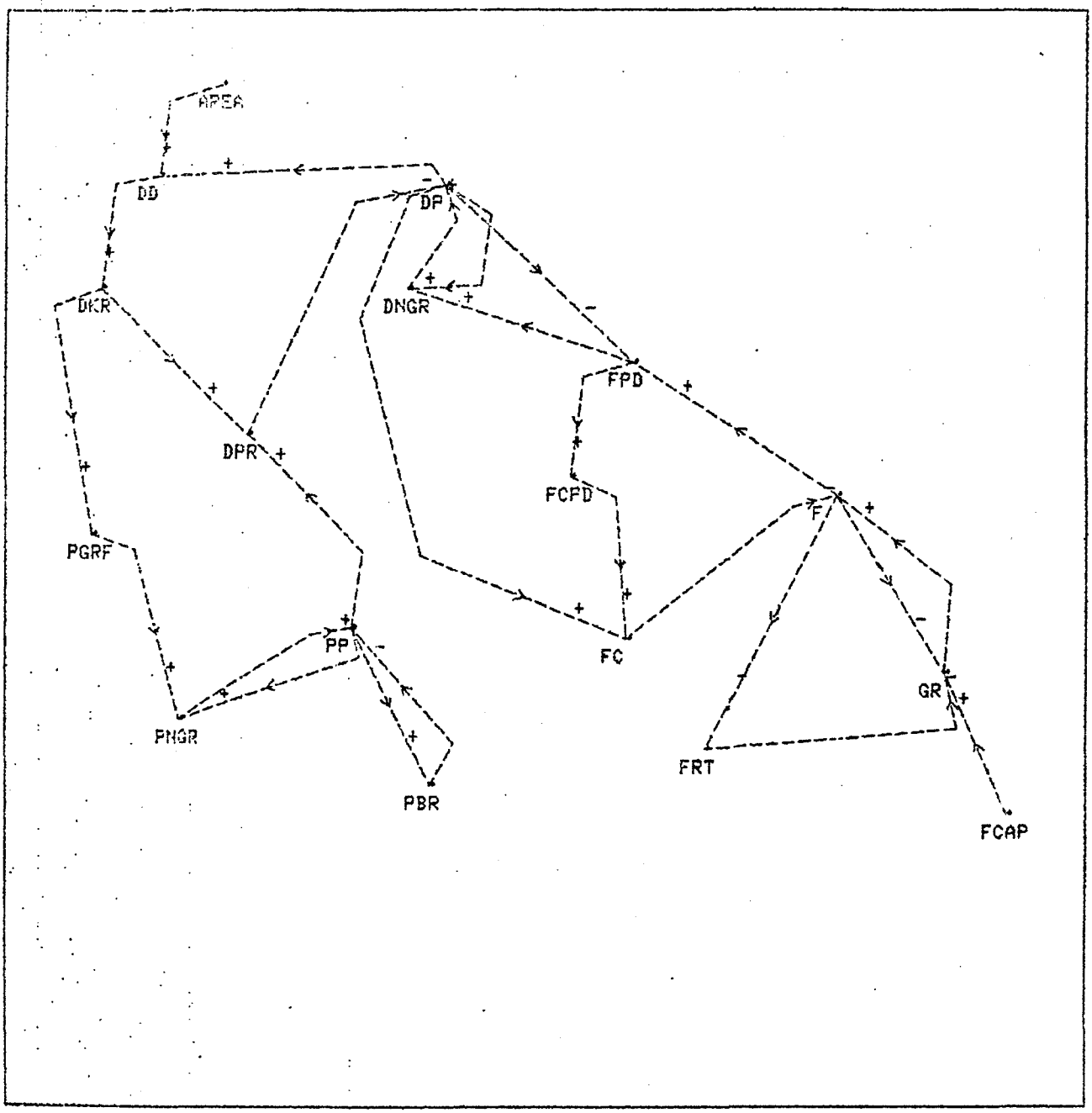


Fig 4



KAIBAB Plateau model

(M. R. GOODMAN)

Fig. 5

DD	DKR	PNGR	PP	PGRF	PBR		

DD	DKR		

Fig. 6

```
        ** RESULTAT TRANSFORMATION **
           ************************

 NIVEAU

DP        F         PP

 TAUX

DPR       DNGR      FC        GR        PNGR      PBR

 AUXILIAIRES

DD        DKR       FPD       FCPD      FRT       PGRF

 INPUT

AREA      FCAP
```

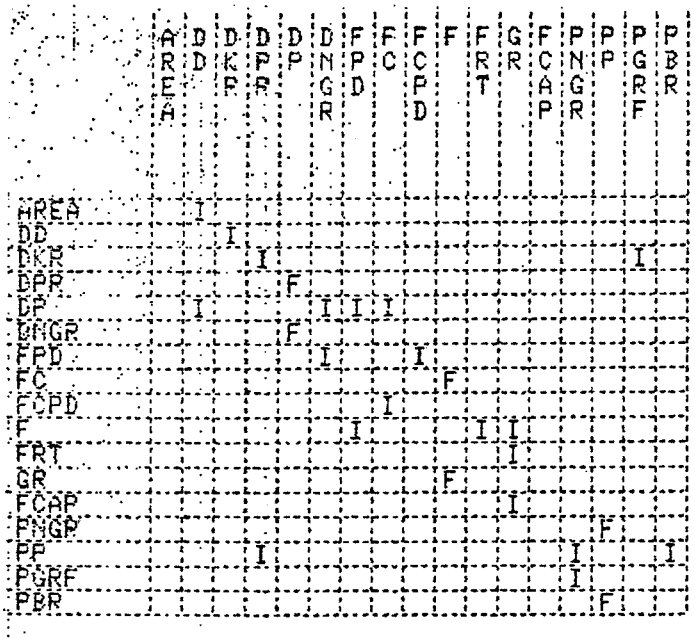
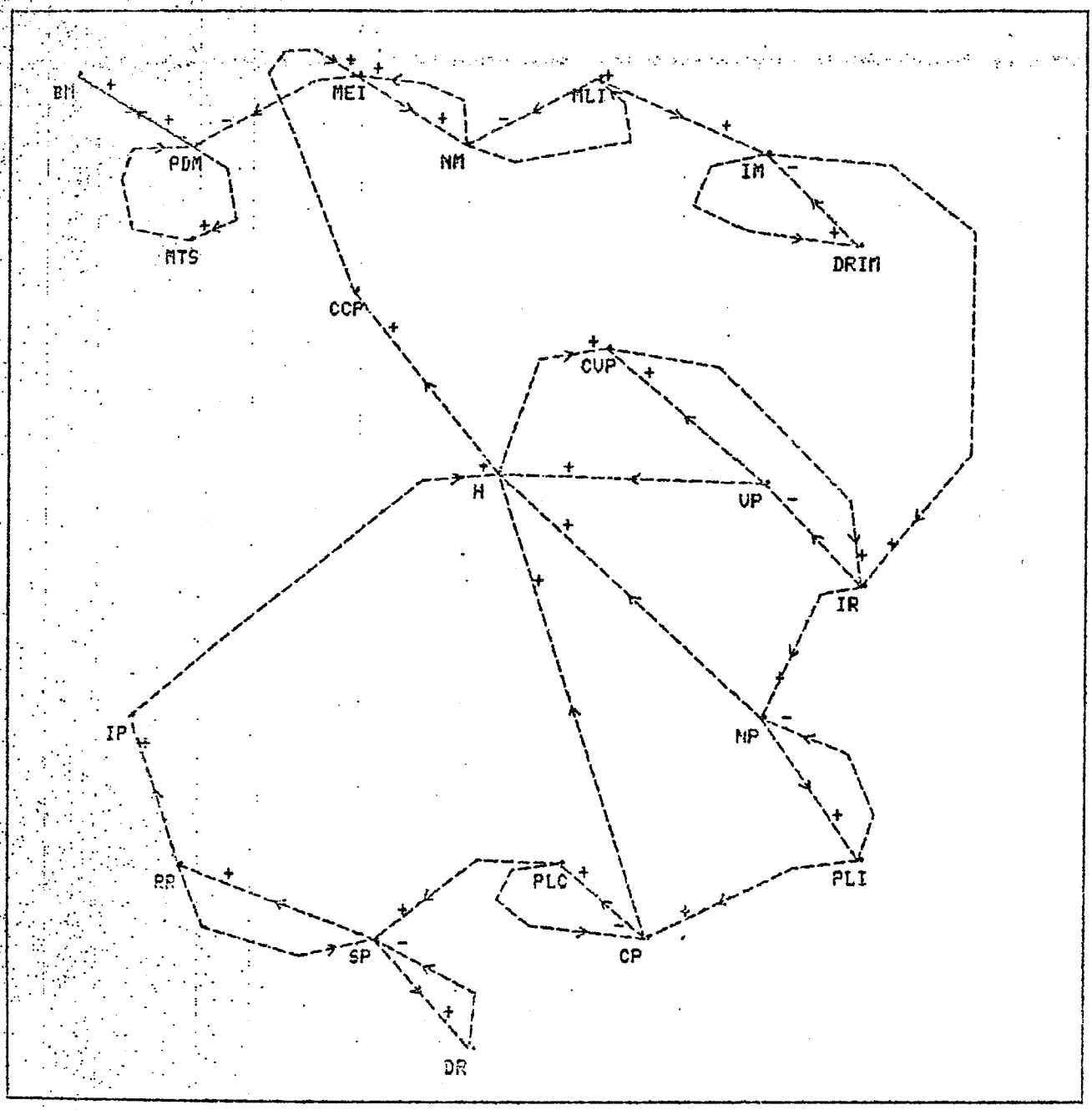


Fig. 7



Yellow fever model

(KJELL KALGRAF)

Fig. 8

** RESULTAT TRANSFORMATION **
****

NIVEAU

PDM NM IM VP NP CP SP IP

TAUX

BM MTS MEI MLI DRIM IR PLI PLC DR RR

AUXILIAIRES

CCP H CVP

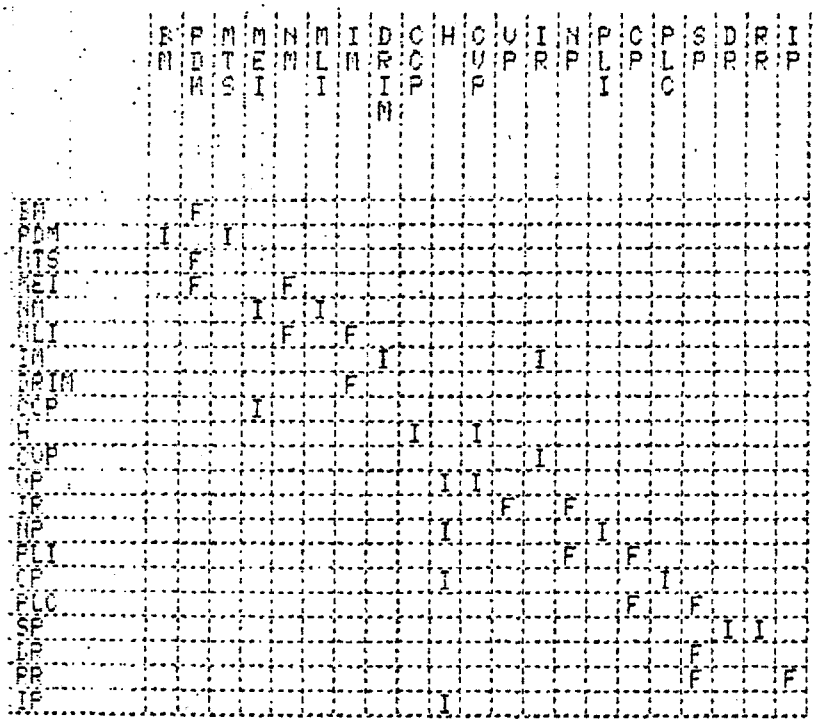
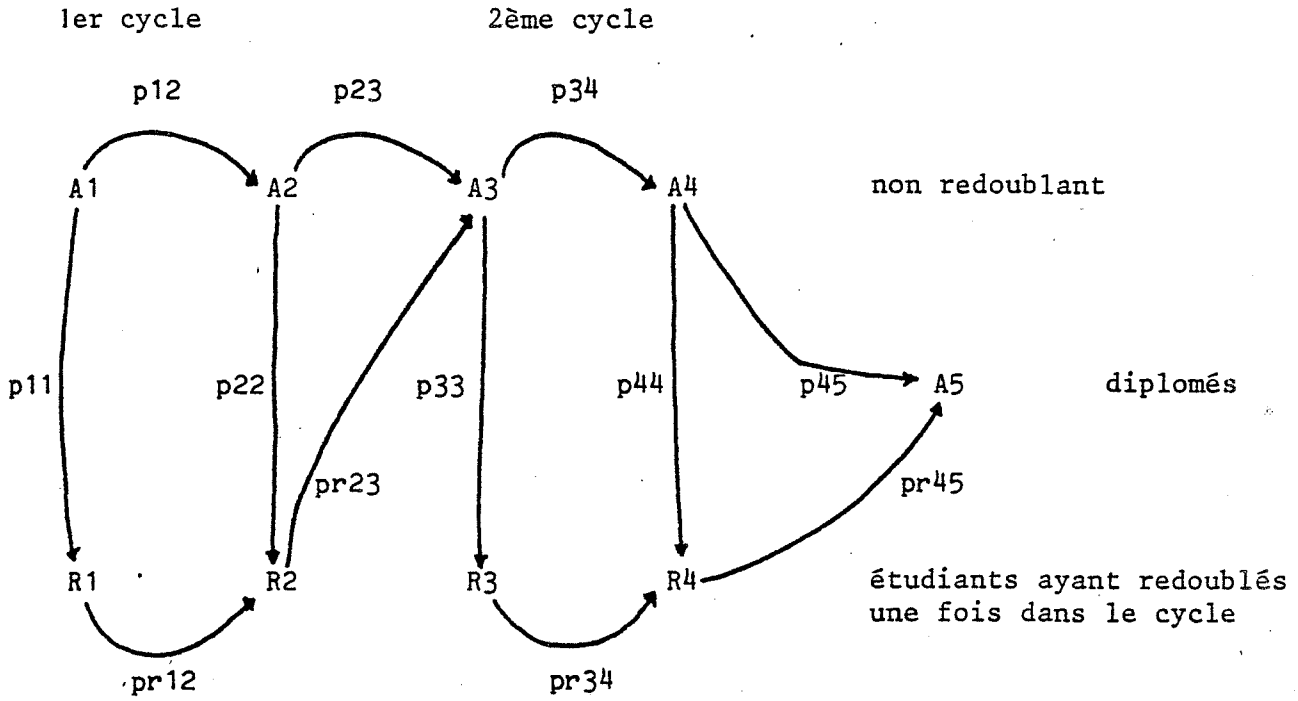


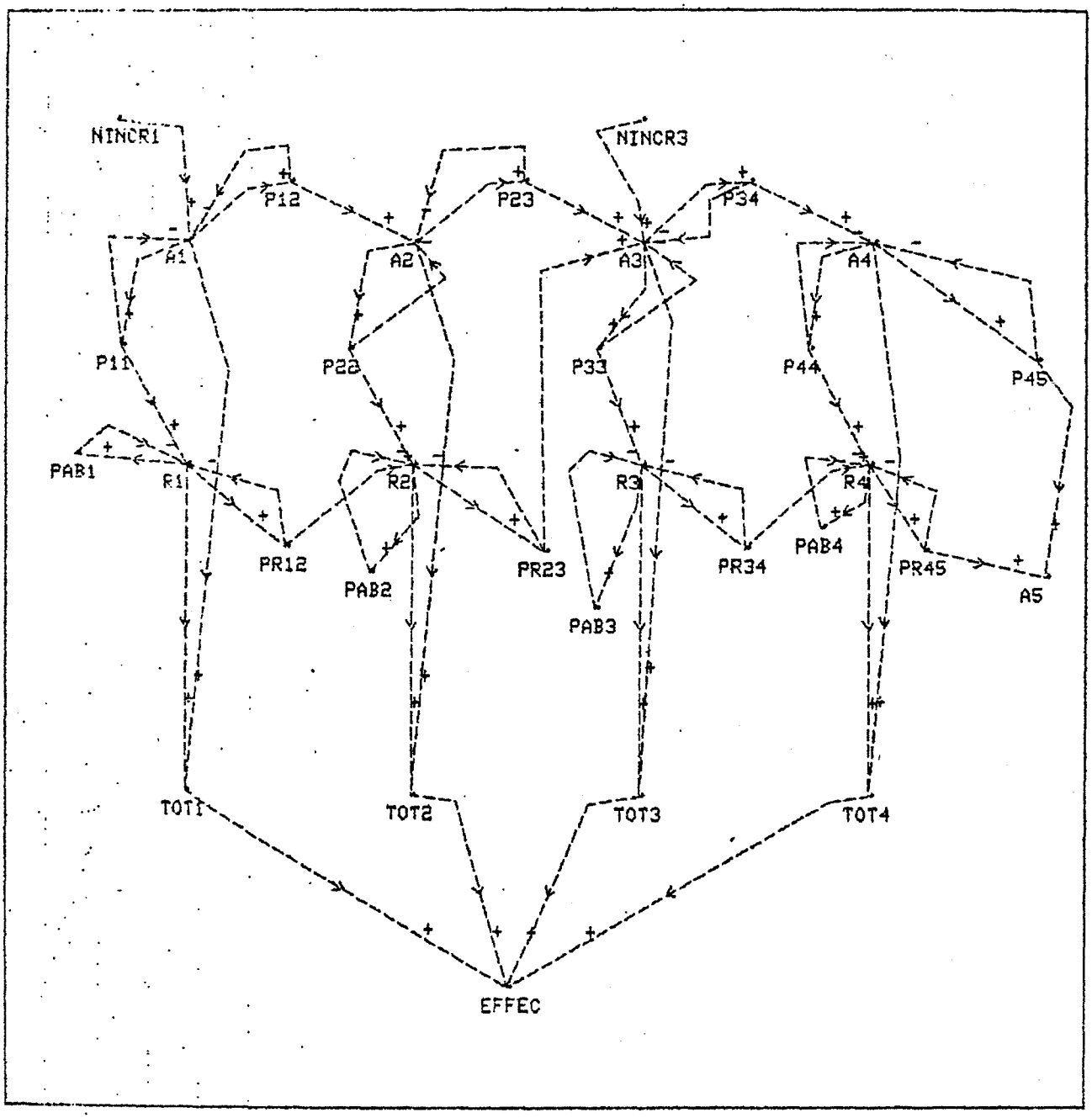
Fig. 9

Le modèle des flux d'étudiants dans une université est une "traduction" d'un modèle markovien en un modèle de la dynamique des systèmes. Le modèle markovien de base est le suivant:



Les pij et les prij étant les probabilités de passer au niveau supérieur (ou de redoubler). Les Ai étant les différentes "années" où ne sont comptés que les étudiants n'ayant pas redoublés. Les Ri les années où sont comptés les étudiants ayant redoublés.

Dans le modèle de dynamique des systèmes des variables ont été rajoutées pour la commodité de l'analyse du comportement du système.



Modèle des flux d'étudiants dans une université

VAYATIS

Fig. 10

```
          ** RESULTAT TRANSFORMATION **
             **********************


   NIVEAU

   P11       P22       P33       P44       P12       P23       P34       P45       PR12      PR23
   PR34      PR45
    TAUX

   A1        A2        A3        A4        R1        R2        R3        R4

    INPUT

   NINCR1    NINCR3

    OUTPUT

   A5        EFFEC
-VOULEZ VOUS LE GRAPHE ? (O/N)
```

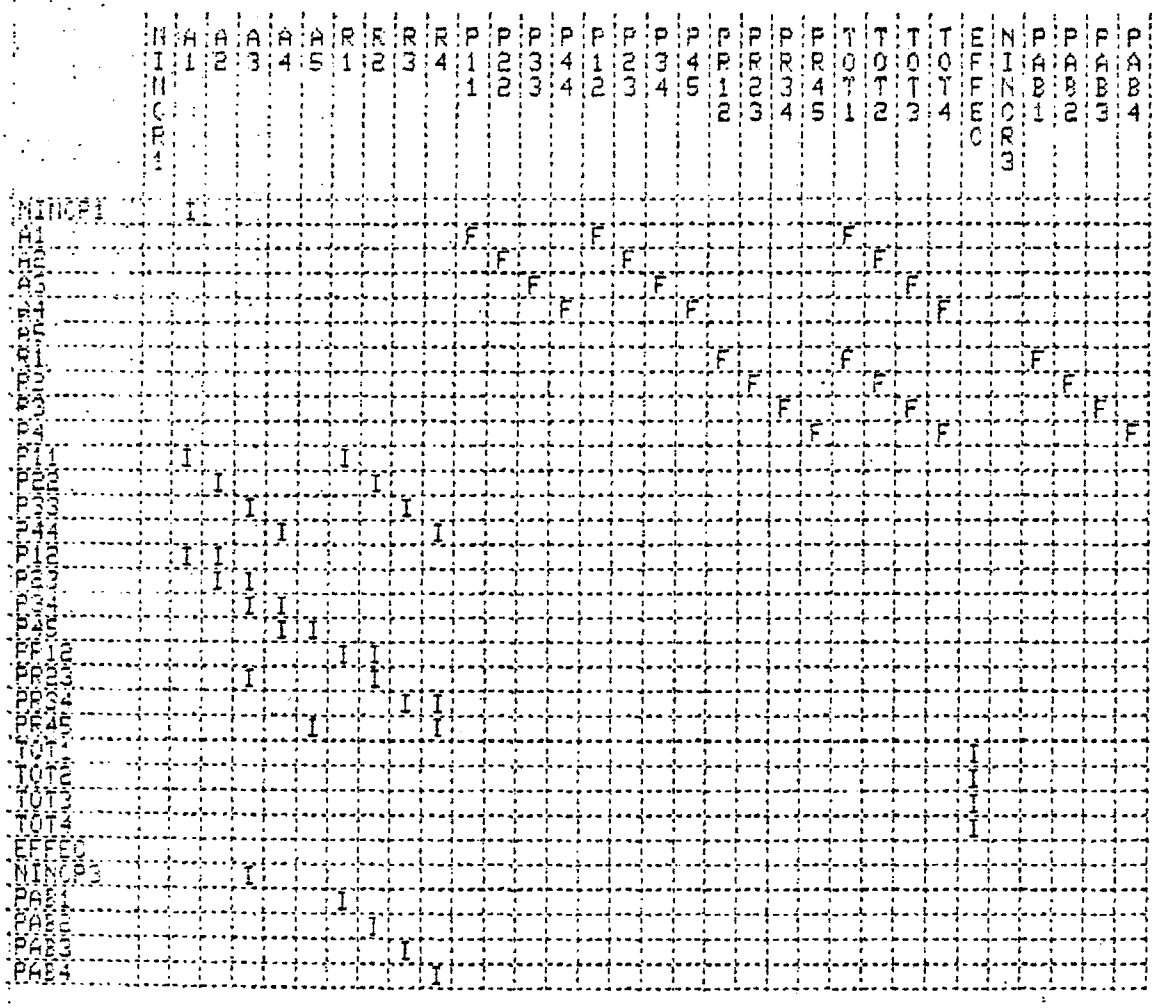


Fig. 11

IV Conclusion et remarques diverses

Après un certain temps d'utilisation du système, des remarques peuvent être faites. Il faut d'abord indiquer que deux directions principales se présentent quand on veut améliorer les possibilités offertes par un système de simulation:

- soit l'amélioration du langage en y ajoutant des concepts nouveaux ou en le remodelant complètement pour en refaire un totalement différent. C'est une voie que de nombreuses personnes ont choisi.

- soit le parti que nous avons pris: améliorer l'environnement de l'utilisateur pour le décharger de toute une partie des tâches techniques qui l'éloignent de sa sphère d'intérêt.

Il est évident que ces deux directions ne sont pas incompatibles, et maintenant que le noyau du système est construit, rien n'empêche d'y intégrer des améliorations du langage. Avant de citer celles qui nous semblent indispensables remarquons tout de même qu'une démarche de recherche d'une plus grande puissance du langage part de l'à priori qu'un modèle sera d'autant meilleur qu'il intègre des aspects de plus en plus fins du système modélisé. Il s'agit donc de donner la possibilité de construire des modèles de plus en plus grands et complexes prenant en compte de plus en plus de détails. Cette attitude, nous l'avons déjà dit, nous semble perverse pour plusieurs raisons:

- en général, on construit un modèle pour simplifier la réalité pour mieux dominer le système étudié. Il est donc contraire à cette démarche de construire un modèle dont le degré de complexité se rapproche de celui du système étudié.

- la modélisation d'un système en prenant en compte les faits microscopiques pour expliquer le comportement macroscopique apporte souvent des déboires. Un même comportement peut exister avec différentes structures microscopiques, donc un luxe de détails peut être une complication inutile et même nuisible.

Il nous semble donc que la meilleure justification d'une amélioration du langage est le confort de l'utilisateur et non une plus grande puissance. Cette amélioration ne doit pas en outre oblitérer l'accès facile au langage pour des utilisateurs non familiés de l'informatique

Parmi les améliorations possibles citons la possiblité de définir des "macros", des sous-modèles, la possiblité de définir des variables vectorisées et d'utiliser des opérateurs globaux de type APL. Ce dernier changement consiste en fait à modifier le langage de base des compilateurs de type DYNAMO, qui est le FORTRAN, en un langage plus évolué de type APL qui présente l'avantage de considérer les entités globalement et non plus éléments par éléments. Une autre amélioration est d'introduire des instructions conditionnelles plus simples et plus explicites que les fonctions qui ont le même usage dans DYNAMO. L'introduction de telles instructions offre une solution partielle au problème de la restructuration dynamique du modèle au cours de la simulation. Une véritable possibilité de restructuration dynamique ne serait acquise qu'au prix d'un métalangage de simulation englobant le modèle et qui le modifierait au gré d'événements significatifs. Cette restructuration pouvant entrainer des changements dans l'ordre d'évaluation des équations, nous voyons que tout cela nécessite des procédures longues et compliquées.

Des améliorations précédentes qui ont trait au langage lui-même nous ne pensons pas tenir compte dans un proche avenir. A part quelques améliorations mineures telles que l'introduction de nouvelles fonctions, nous ne nous intéresserons pas à cet aspect des choses. Nous mettrons plutôt l'accent sur l'environnement de l'utilisateur et son confort dans la mise au point des modèles.

Nous pensons que pour qu'un utilisateur non porté sur l'informatique puisse utiliser ce système avec plus de facilité, il sera nécessaire d'améliorer le dialogue en ajoutant une fonction d'aide, appelable à tous les niveaux, qui indiquera plus en détail que ne le fait la procédure habituelle, les choix possibles et leur signification tout en les situant dans leur contexte d'utilisation. L'idéal est de remplacer le manuel d'utilisation par le système lui-même. Les personnes familières du système ne devront pas être gênées par ce dialogue plus long qui ne devra apparaitre qu'à la demande.

Ce premier point précisé, il nous semble qu'au niveau de la mise au point il faudra ajouter quelques outils permettant de meilleurs diagnostics. L'un de ces outils nous parait être l'introduction de points d'arrêt conditionnels qui stopperont la simulation sur certaines conditions définies au départ et permettant de vérifier le comportement du modèle en cours de simulation, éventuellement de modifier des paramètres puis de relancer la simulation ou de l'arrêter définitivement. L'intérêt de ces points d'arrêt est plus grand que le simple fait d'être une aide à la mise au point: on pourra s'en servir pour piloter le modèle et tester ainsi diverses politiques. Nous transformons ainsi, dans une certaine mesure, un outil de simulation en un "jeu", au sens des jeux d'entreprises, où une personne intervient dans le comportement du modèle. Une extension de ces points d'arrêt peut aussi être un contrôle de la durée de simulation non plus sur la date de fin mais sur un événement quelconque défini préalablement ou une combinaison d'événements, la date n'étant plus qu'un événement parmi d'autres.

D'autres ajoûts pourraient être la définition de "macro d'éxecution" qui permettraient de lancer l'éxecution d'un même modèle n fois en faisant varier divers paramètres pour obtenir plusieurs jeux de résultats en une fois.

Comme on le voit, les améliorations ou les gadgets sont nombreux et la liste présentée est loin d'être exhaustive. Certains de ces gadgets s'avéreront inutiles, d'autres nécessaires, c'est pourquoi un tel système d'aide à la modélisation se doit d'être évolutif et d'accepter les modifications éventuelles.

D'un autre côté, plus généralement qu'a apporté ce travail? Nous pensons pouvoir tirer plusieurs conclusions au vu des résultats. D'une part le fait d'avoir opté pour un mode interprété pour la phase de mise au point des modèles permet de gagner un temps appréciable en évitant les phases de compilation et d'édition de liens toujours longues et usantes pour les nerfs. Ces deux phases prennent en effet un temps supérieur à ce que le mode interprété fait perdre à l'éxecution par rapport au mode compilé. D'autre part une technique qui semble économiser du temps et du travail, là aussi usant, est l'éditeur de texte spécialisé qui signale les erreurs de syntaxe immédiatement. Même si cela n'est pas toujours possible, certaines erreurs ne pouvant pas se détecter sans une analyse globale du programme, il semble que cela soit une solution partielle au problème de la construction des programmes. Il est évident qu'une grosse part des erreurs doivent être détectées "manuellement" celles-ci étant du ressort d'un traitement plus sémantique que syntaxique.

Au niveau de la construction graphique du modèle, le nombre d'équations générées automatiquement fait que dans ce cas précis cette technique est un peu décevante. Mais il semble qu'avec d'autres langages où le nombre de symboles graphiques est plus grand et où donc l'ambiguité pour la définition des équations est levée, elle soit une technique d'avenir. Des langages tels que THTSIM (Bond graph) pourraient être mieux adaptés et donc présenter des avantages indéniables, cette technique graphique ayant donné des résultats néanmoins séduisants avec un langage moins bien adapté.

Annexe : Exemple d'utilisation.

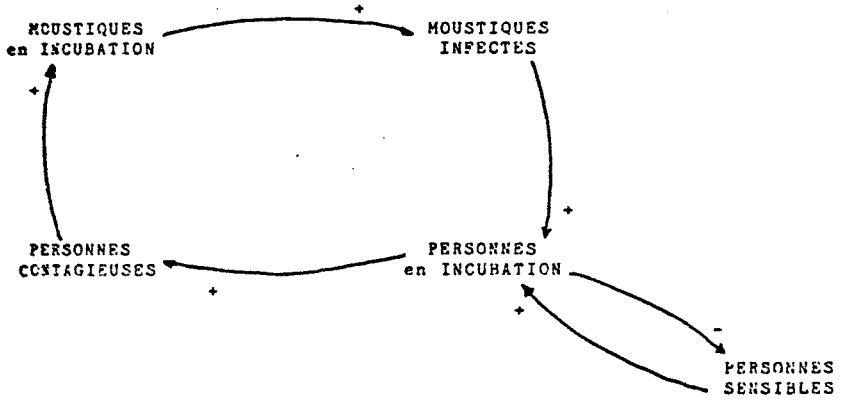
Nous allons maintenant prendre un exemple pour voir les possibilités de Dynamine. L'exemple choisi, le modèle d'une épidémie de fièvre jaune, est tiré du livre de GOODMAN, STUDY NOTES IN SYSTEM DYNAMICS.C'est un modèle relativement simple qui montre quelques unes des possibilités du système tout en ne nécessitant pas de longs développements.

Le but du modèle est de vérifier une hypothèse de propagation de l'épidèmie en comparant le comportement du modèle avec la réalité: une épidémie de fièvre jaune est caractérisée par une brusque augmentation du nombre de malades suivie d'une baisse aussi rapide.

L'hypothèse est la suivante:

> des moustiques infectés piquent des personnes saines et leur transmettent la maladie. Ces personnes deviennent contagieuses pendant un laps de temps déterminé. Si un moustique non infecté pique une de ces personnes pendant la période de contagion, il devient porteur de la maladie. Les personnes qui réchappent de la maladie sont immunisées à vie.

Le diagramme causal simplifié est donc le suivant:



Nous allons donc étudier ce modèle. Pour cela, nous procéderons en diverses étapes:

- construction du diagramme causal
- détermination du type des variables
- construction du diagramme des flux
- écriture des équations
- exécution
- étude des résultats

Toutes ces étapes sont effectuées par le programme DYN.

Ce programme pose les questions nécessaires au fur et à mesure de la progression de l'utilisateur.

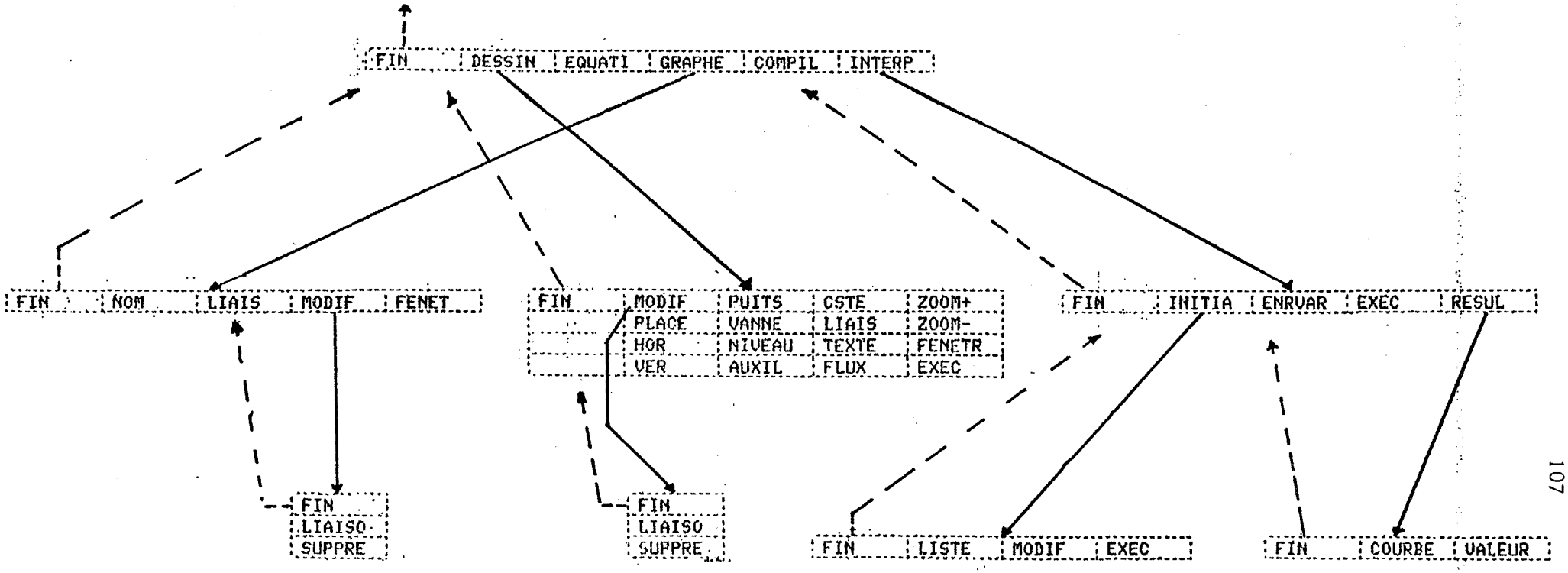
Au démarrage la première question est la suivante:

```
DYNAMINE     EDITEUR

NOM DU PROGRAMME ? (6 CARACT. )
```

Puis apparait le menu du premier niveau (voir figure 1 pour les différents niveaux)

```
| FIN | DESSIN | EQUATI | GRAPHE | COMPIL | INTERP |
```



Cheminements possibles à travers les différents menus

Fig. 1

A Diagramme causal

Pour construire le diagramme causal nous désignons GRAPHE. Puis dans le menu suivant:

FIN	RELAT	STRUCT

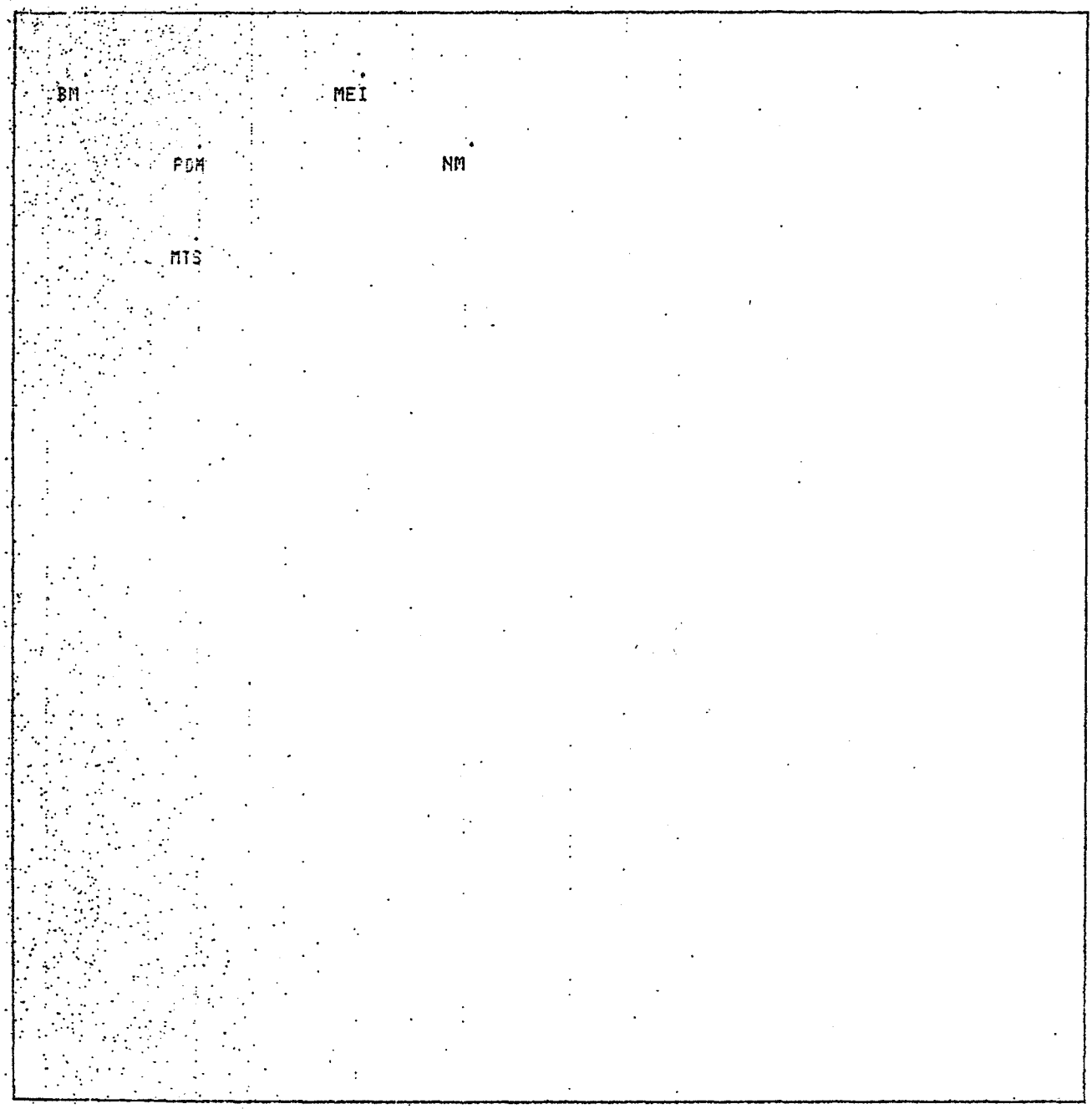
Relat qui permet de construire graphiquement le diagramme. Pour cela nous procédons de la façon suivante:

Dans le menu

FIN	NOM	LIAIS	MODIF	FENET

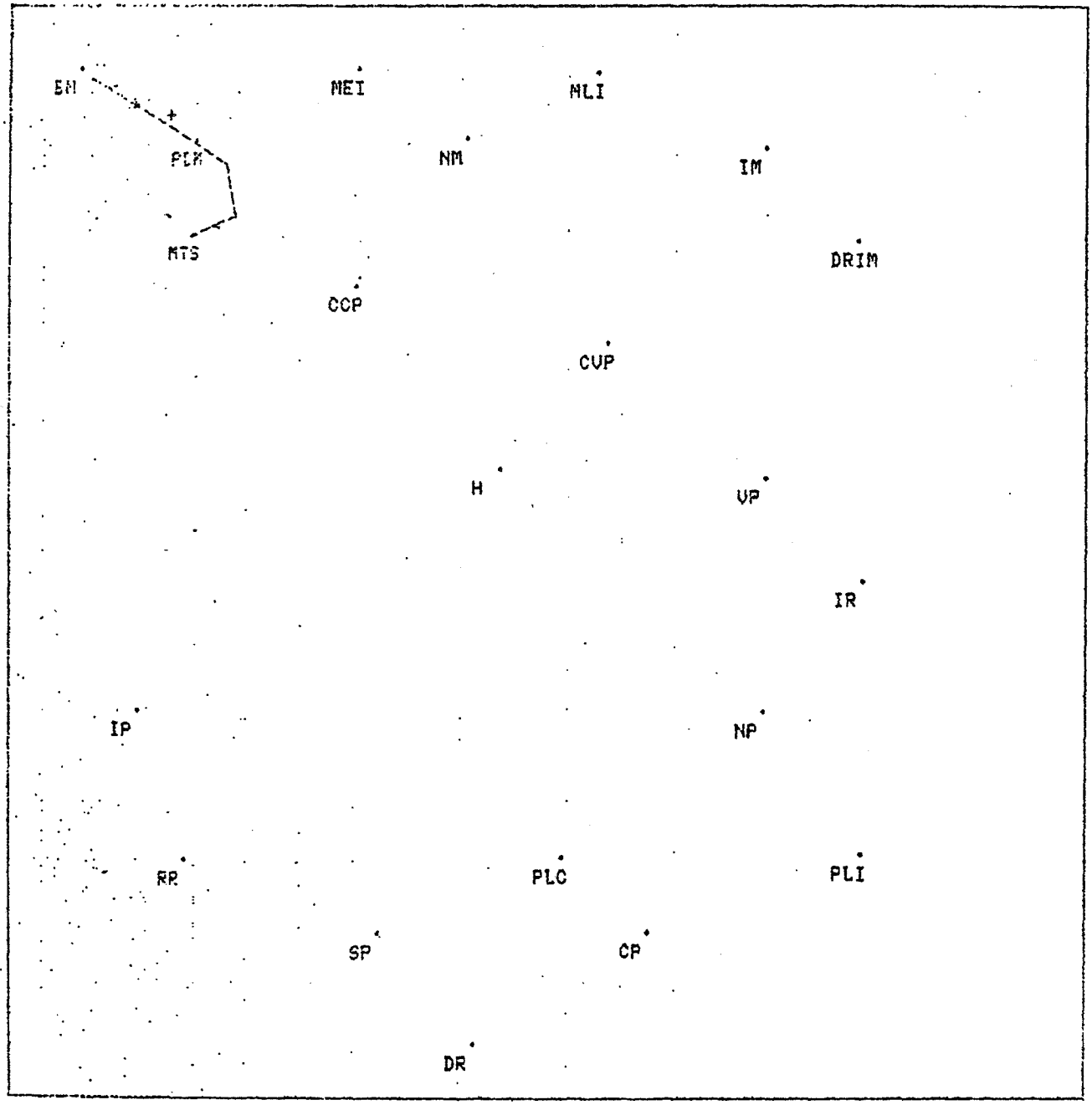
nous désignons NOM pour entrer les noms des variables et les placer dans le plan support du diagramme. Puis nous désignons LIAIS pour décrire les liaisons. Ces liaisons sont enregistrées en désignant la variable départ, les points par lesquels elles passent, puis la variable arrivée. Pour le signe il suffit de taper un - ou un + en désignant la variable arrivée, il se placera automatiquement à proximité de la liaison.

Différentes étapes sont visibles sur les figures 2 à 4



Implantation de variables Fig. 2

L'utilisateur tape le nom puis indique au réticule où le placer.

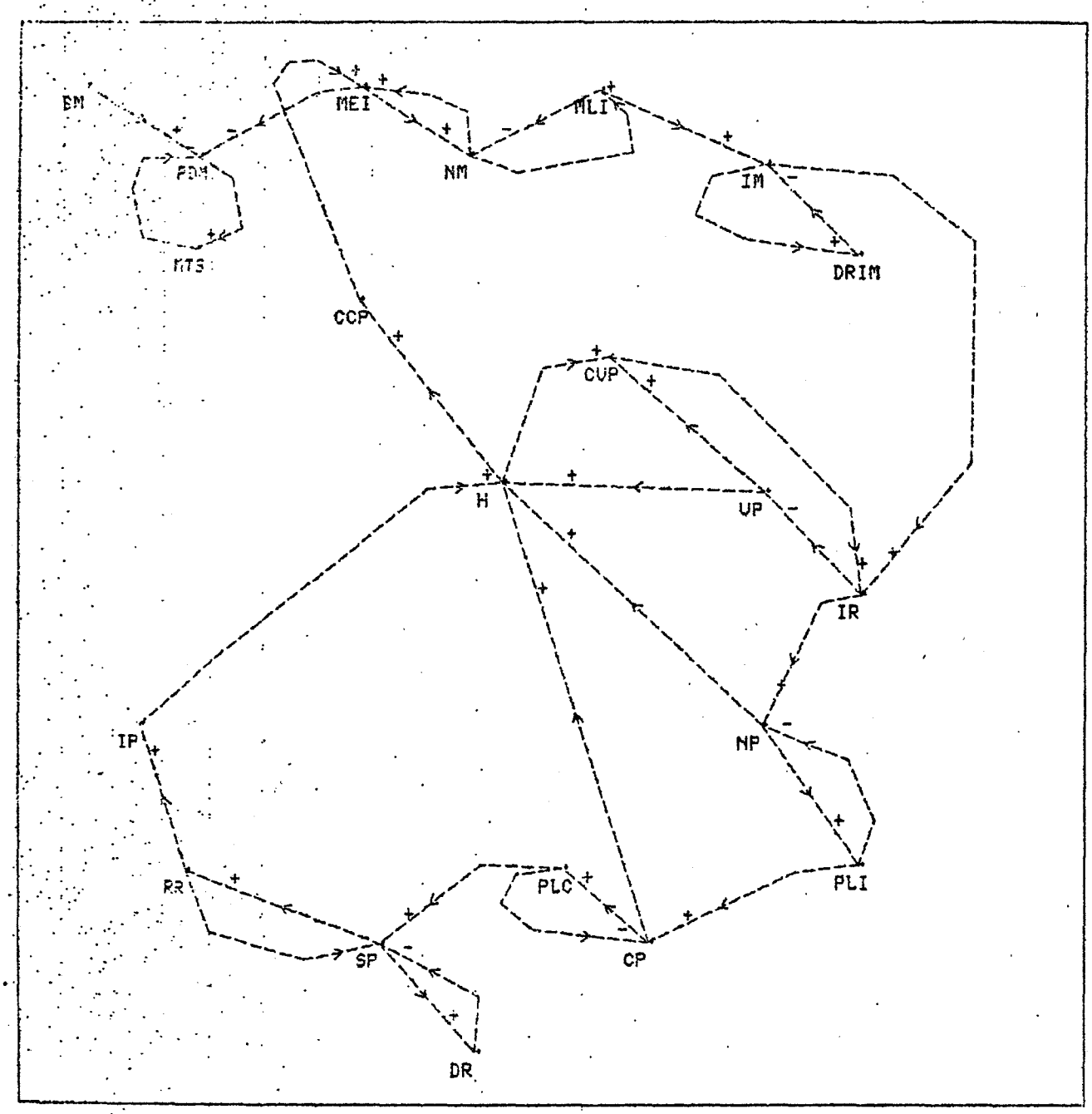


Création de liaisons

L'utilisateur pointe la variable origine, les points par lesquels la liaison doit passer, puis la variable extrémité.

Fig. 3

FIN | NOM | LIAIS | MODIF | FENET



Réseau complet

Fig. 4

Quand le diagramme est fini, nous repassons au niveau supérieur en désignant FIN, et nous passons à l'étape détermination du type des variables en désignant STRUCT dans le menu

FIN | RELAT | STRUCT

L'algorithme de BURNS employé dans l'exemple ne s'arrête qu'en cas d'ambiguité et présente dans un menu la liste des variables dont le type ne peut être déterminé. Ces différents points sont présentés sur la figure 5 . Dans le cas présent, trois interventions ont été nécessaires. Elles ont consisté à désigner

1) PDM comme niveau grâce au menu

2) VP comme niveau

3) CCP comme auxiliaire

Ce qui a donné les résultats de la figure 6

BM	PDM	MTS	MEI	NM	MLI	IM	DRIM	CCP	H	CVP	VP	IR	NP	PLI
CP	PLC	SP	DR	RR	IP									

CCP	H	CVP	VP	IR	NP	PLI	CP	PLC	SP	DR	RR	IP		

CCP	CVP

Fig. 5

** RESULTAT TRANSFORMATION **
****

NIVEAU

PDM NM IM VP NP CP SP IP

TAUX

BM MTS MEI MLI DRIM IR PLI PLC DR RR

AUXILIAIRES

CCP H CVP

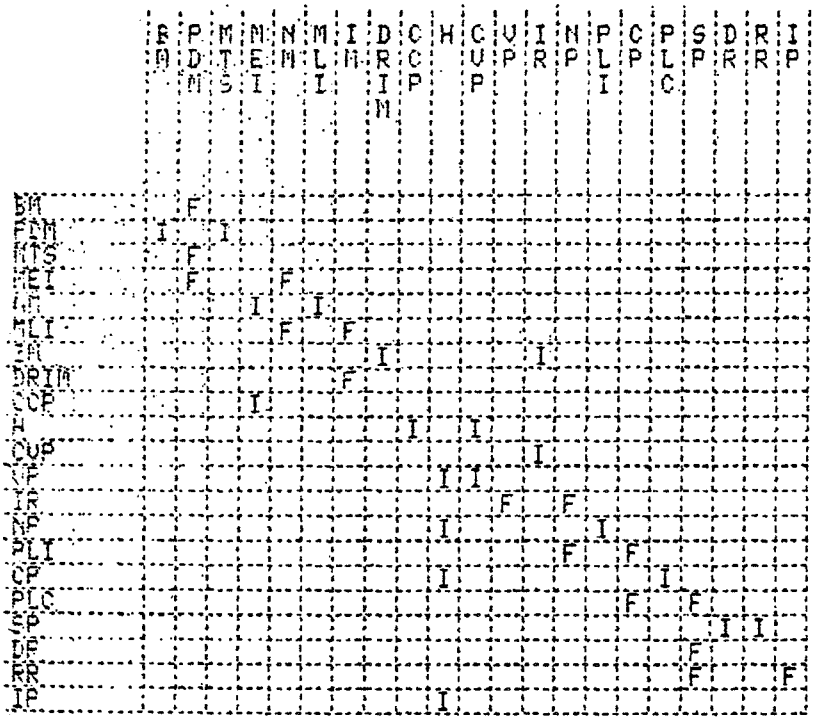


Fig. 6

B. Diagramme des flux

Pour construire le diagramme des flux, nous désignons DESSIN dans le menu

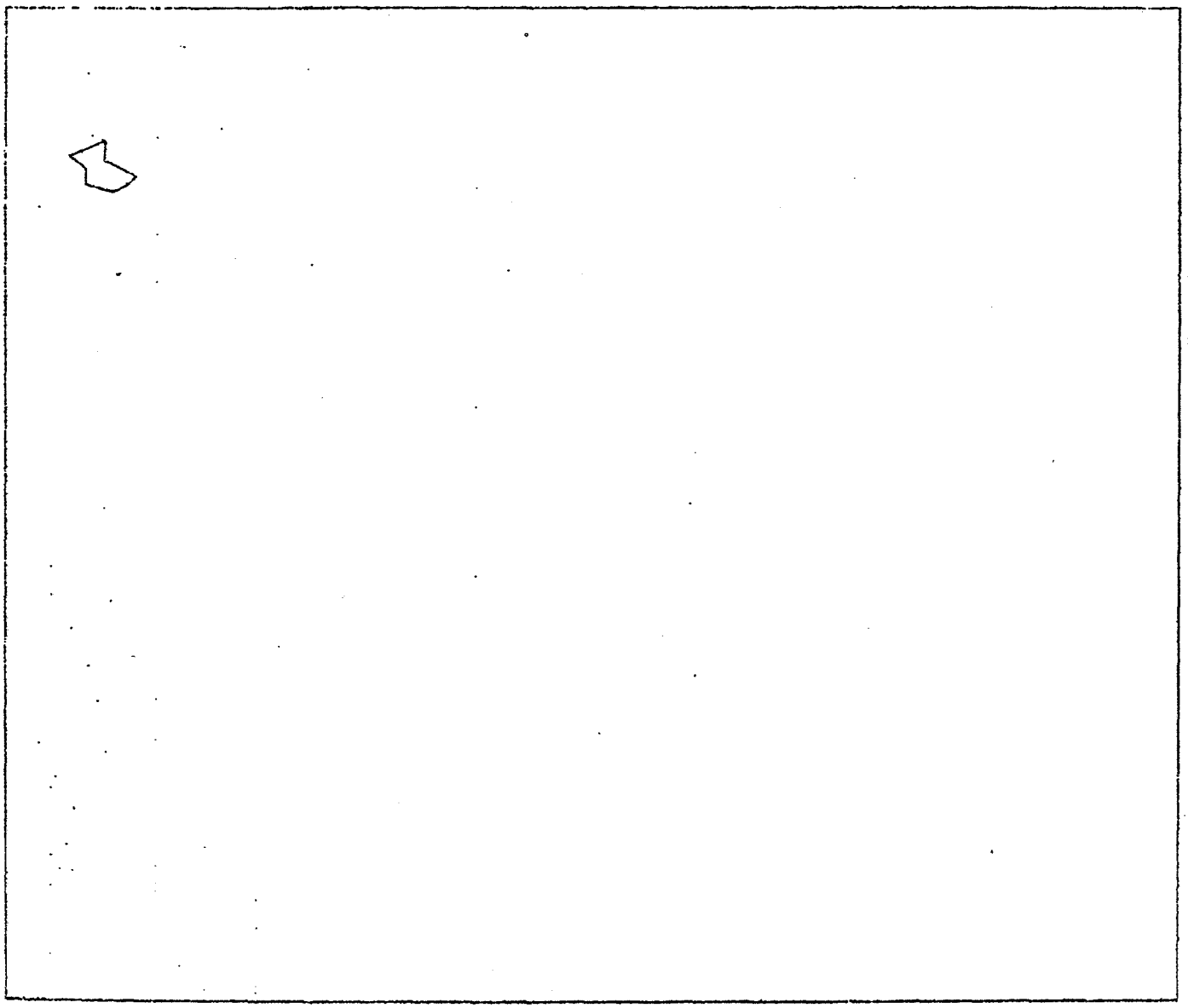
FIN	DESSIN	EQUATI	GRAPHE	COMPIL	INTERP

Sur l'écran apparait le menu et le cadre de la figure 7, qui permettront la construction.

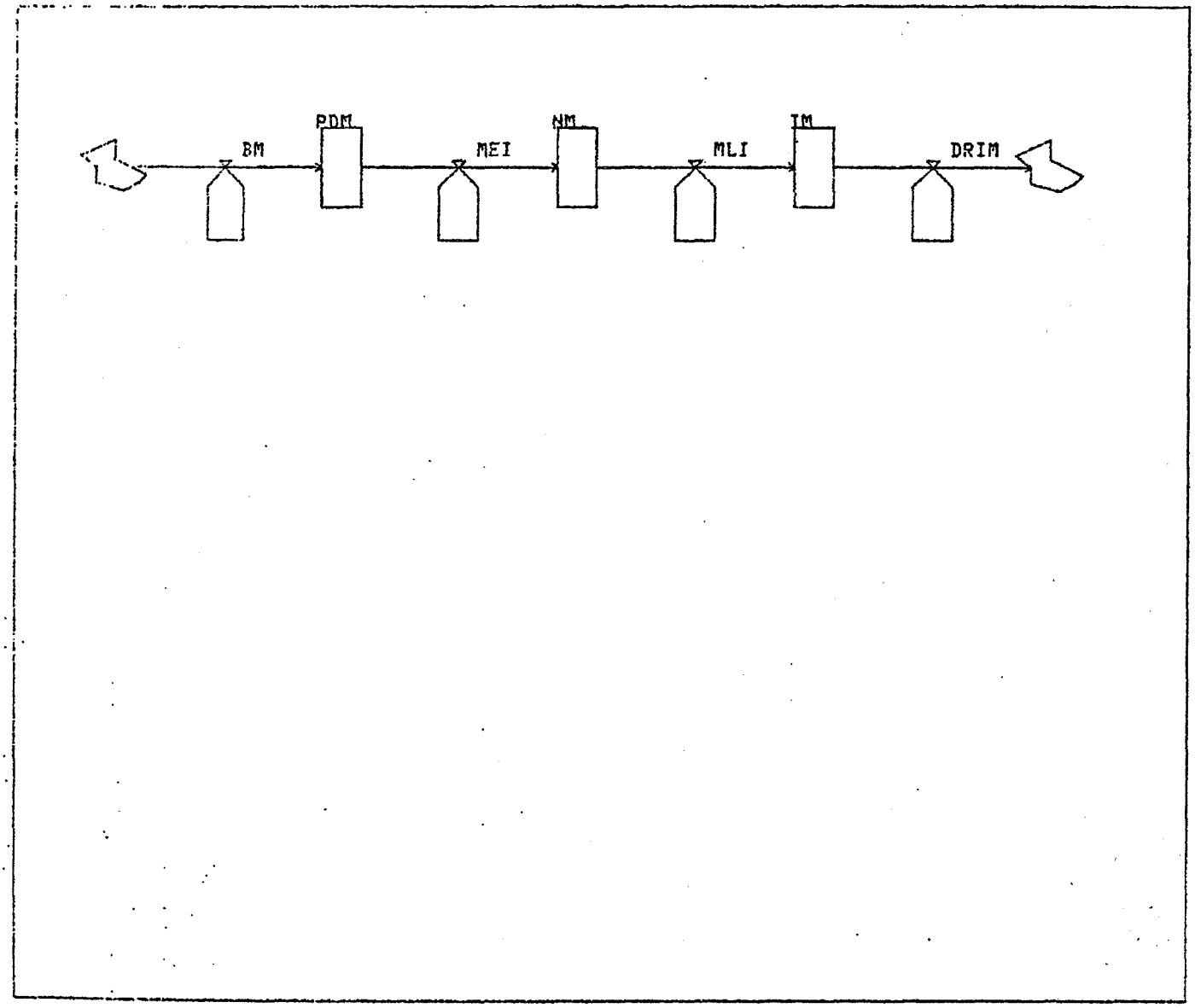
Pour construire le diagramme, nous plaçons d'abord un puits, départ du premier réseau. La position de ce puits conditionne l'emplacement de toute la suite du réseau. Puis nous désignons successivement VANNE et NIVEAU dans le menu pour construire la suite, tout en donnant les noms des variables correspondantes. (figures 8 à 13)

La suite des figures présente diverses étapes de la construction, dont une modification de la place d'une vanne. (figures 10 et 11)

Le résultat est le diagramme de la figure 13.



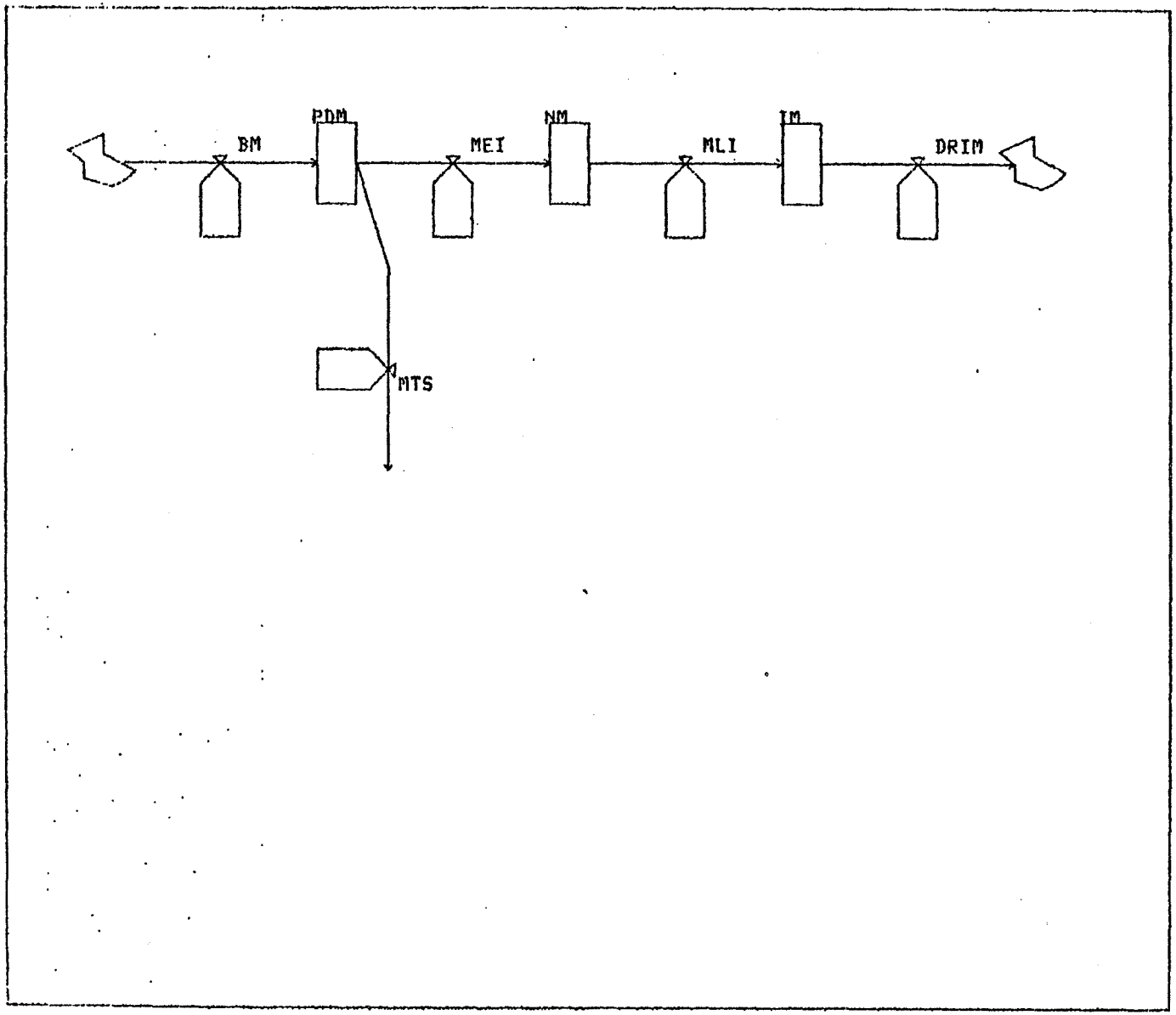
Placement du puits origine



Mise en place d'un réseau

Fig. 9

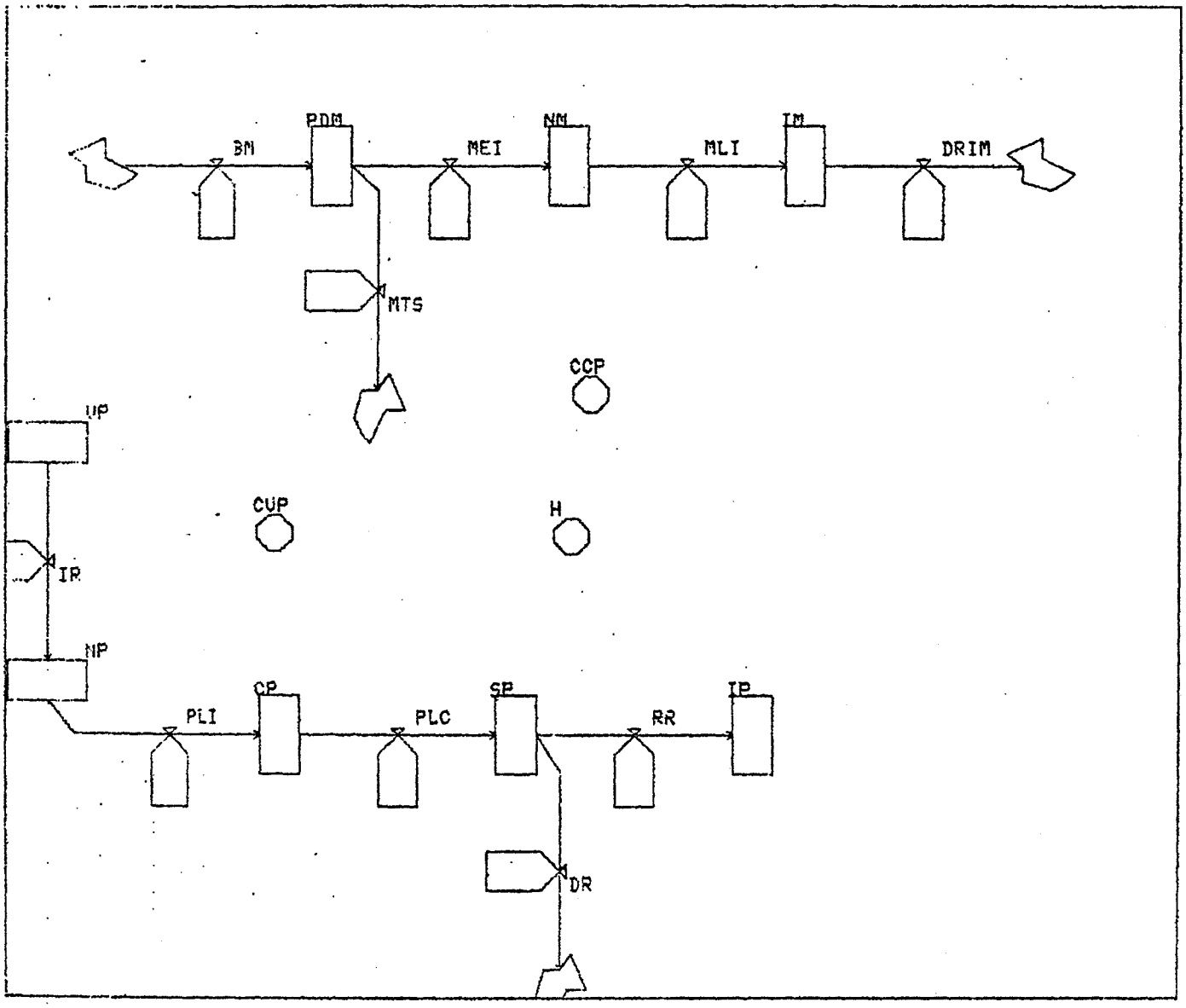
FIN	MODIF	PUITS	CSTE	ZOOM+
	PLACE	VANNE	LIAIS	ZOOM-
NOM ?	[illegible]	NIVEAU	TEXTE	FENETR
MTS	[illegible]	AUXIL	FLUX	EXEC



FIN
LIAISO
SUPPRE

La vanne MTS va être déplacée.

Fig. 10



Réseau en construction. On voit ici l'effet du fenêtrage.

Fig. 11

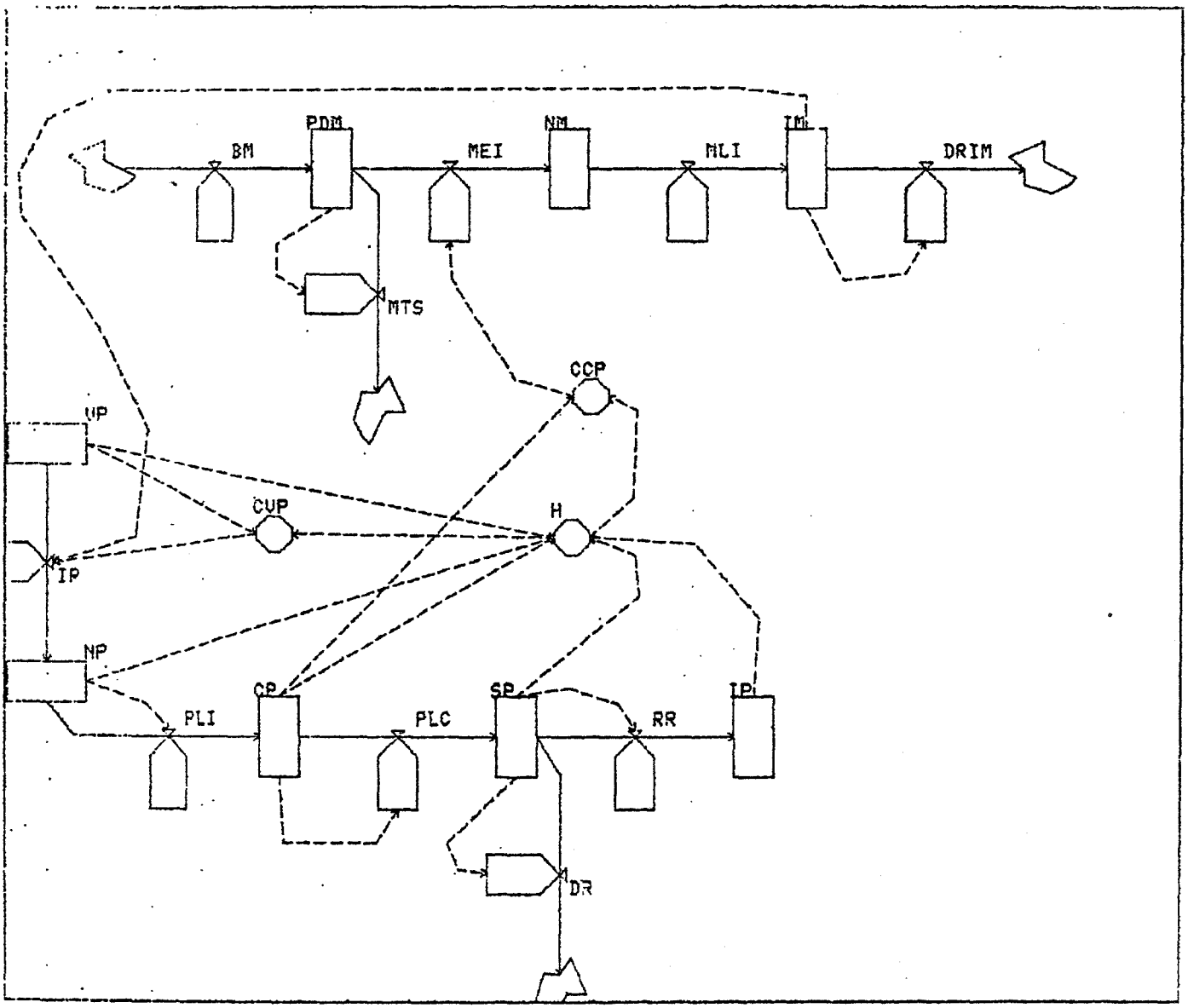
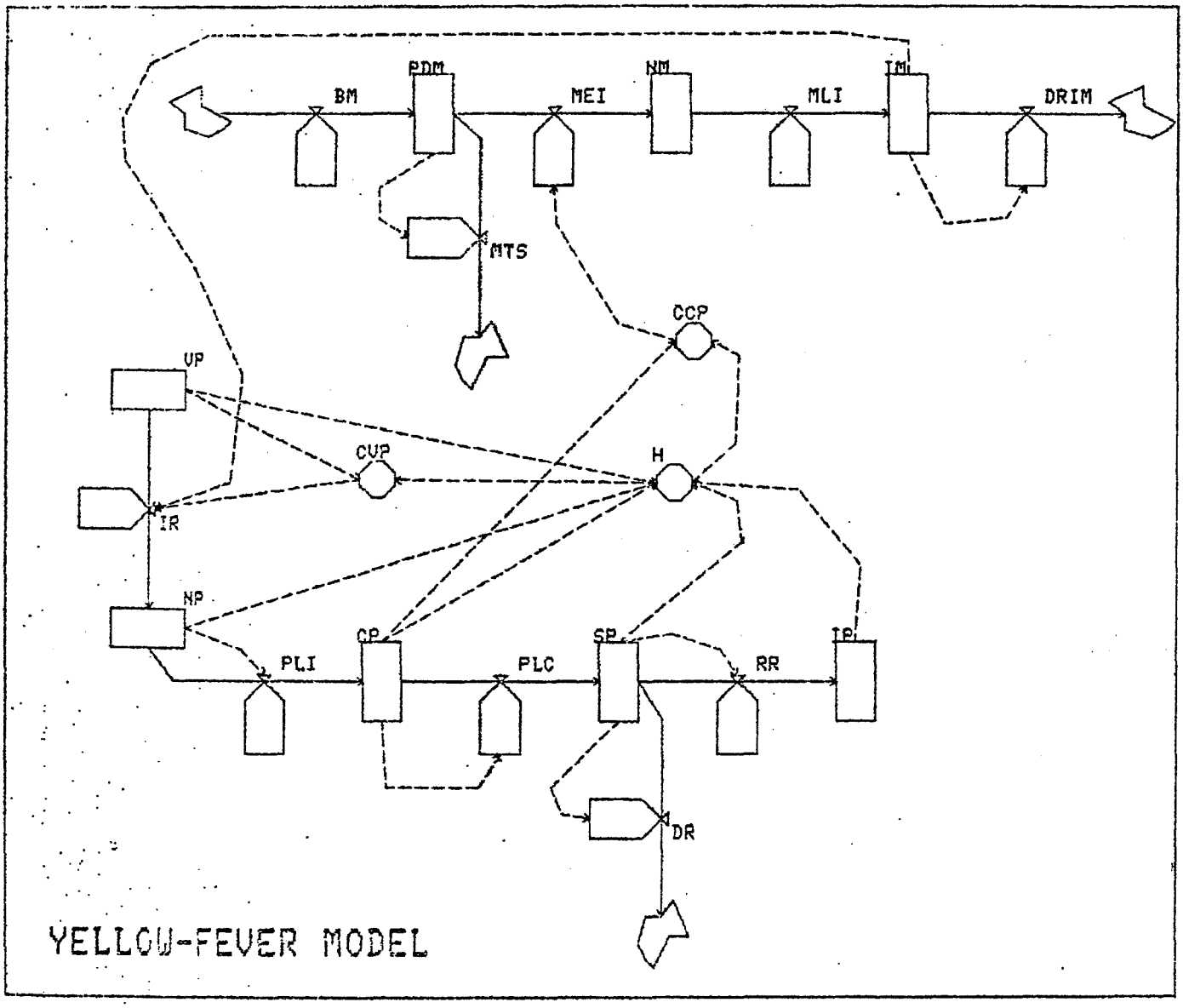

BM
PDM
MEI
NM
MLI
IM
DRIM
MTS
CCP
UP
CUP
H
IP
NP
CP
SP
IPI
PLI
PLC
RR
DR

Fig. 12



Réseau complet. Le diagramme a été recentré par un déplacement de la fenêtre.

Fig. 13

C. Ecriture des équations.

Cette étape est atteinte en désignant EQUATI dans le menu

FIN	DESSIN	EQUATI	GRAPHE	COMPIL	INTERP

Les choix possibles sont présentés par

FIN	TABLE	LISTE	EQUATI	NOMVAR

EQUATI	écriture des équations
NOMVAR	ajoût d'une nouvelle variable sans repasser par une autre étape.
LISTE	liste des équations. La figure 14 en donne un exemple avant toute écriture, ce qui permet de voir quelles équations ont été générées automatiquement.
TABLE	entrée des valeurs d'une table
FIN	retour au niveau supérieur

Pour écrire les équations nous désignons donc EQUATI. Le menu comportant la liste des noms de variable est alors affiché, qui permet de désigner celle dont on veut écrire l'équation. Cette équation est écrite au moyen de l'éditeur spécialisé qui comporte le menu de la page suivante.

+	-	*	/	**	(J)	(K)	DT	(	)	,	=			
DELAI	ABS	FLOAT	AINT	AMOD	AMAX1	AMIN1	SIGN	DIM	EXP	ALOG	ALOG10	SQRT	SIN	COS
TANH	ATAN	RAN	P3SUP4	PA3ZER	TABLE	TABLEH	ESCALI	SAUT	ALEAT	ALEASM	PULSAT	LISINF		
INSERT	SUPPRE	INIVAR	INIPAR	FIN										
BM														

BM(K)=M/MT

On peut écrire les équations soit au clavier, soit en désignant les éléments du menu au réticule.

La liste des équations est en figure 15.

```
LISTING  EQUATIONS DU MODELE

 BM(K)=
PDM(K)=PDM(J)+DT*(+BM(J)-MEI(J)-MTS(J))
 MEI(K)=
NM(K)=NM(J)+DT*(+MEI(J)-MLI(J))
 MLI(K)=
 IM(K)=IM(J)+DT*(+MLI(J)-DRIM(J))
 DRIM(K)=
 MTS(K)=
 VP(K)=VP(J)+DT*(-IR(J))
 IR(K)=
NP(K)=NP(J)+DT*(+IR(J)-PLI(J))
 PLI(K)=
CP(K)=CP(J)+DT*(+PLI(J)-PLC(J))
 PLC(K)=
SP(K)=SP(J)+DT*(+PLC(J)-RR(J)-DR(J))
 RR(K)=
IP(K)=IP(J)+DT*(+RR(J))
 DR(K)=
 CCP(K)=
 CVP(K)=
 H(K)=

POUR LA SUITE TAPER  CR
```

ici on remarque les équations générées automatiquement à partir du diagramme des flux (équations de niveaux). Dans ce modèle, la proportion importante de niveaux rend cette génération particulièrement intéressante.

Fig. 14

```
LISTING  EQUATIONS DU MODELE

 BM(K)=M/MT
PDM(K)=PDM(J)+DT*(+BM(J)-MEI(J)-MTS(J))
 PDM=0
 MEI(K)=BDM*PDM(K)*CCP(K)*NMPB
NM(K)=NM(J)+DT*(+MEI(J)-MLI(J))
 NM=0
 MLI(K)=NM(K)/MIP
 IM(K)=IM(J)+DT*(+MLI(J)-DRIM(J))
 IM=0
 DRIM(K)=IM(K)/MINP
 MTS(K)=PDM(K)/MUP
 VP(K)=VP(J)+DT*(-IR(J))
 VP=20000
 IR(K)=IM(K)*CVP(K)*IPB*BDM
NP(K)=NP(J)+DT*(+IR(J)-PLI(J))
NP=100
PLI(K)=NP(K)/HIP
CP(K)=CP(J)+DT*(+PLI(J)-PLC(J))
CP=0
PLC(K)=CP(K)/COP
SP(K)=SP(J)+DT*(+PLC(J)-RR(J)-DR(J))
SP=0
RR(K)=SP(K)*(1-FD)/SPE
IP(K)=IP(J)+DT*(+RR(J))
IP=0
DR(K)=SP(K)*FD/SPE
H(K)=VP(K)+NP(K)+CP(K)+SP(K)+IP(K)
CCP(K)=CP(K)/H(K)
CVP(K)=VP(K)/H(K)

POUR LA SUITE TAPER  CR
```

Fig. 15

D. Exécution

Pour cela il faut désigner soit COMPIL, soit INTERP. Nous avons choisi INTERP qui évite une longue étape de compilation et d'édition de liens.

La première opération effectuée est le tri des équations. L'ordre d'interprétation des équations est donné en figure 16.

LISTE DES EQUATIONS TRIEES

1	BM	1
2	PDM	2
3	NM	4
4	MLI	5
5	IM	6
6	DRIM	7
7	MTS	8
8	VP	9
9	NP	11
10	PLI	12
11	CP	13
12	PLC	14
13	SP	15
14	RR	16
15	IP	17
16	DR	18
17	H	19
18	CCP	20
19	MEI	3
20	CVP	21
21	IR	10

LISTE DES EQUATIONS D'INITIALISATION

1	PDM	2	2
2	NM	4	4
3	IM	6	6
4	VP	9	11
5	NP	11	9
6	CP	13	14
7	SP	15	16
8	IP	17	18
21	BM	1	1
22	MLI	5	5
23	DRIM	7	7
24	MTS	8	8
25	PLI	12	13
26	PLC	14	15
27	RR	16	17
28	DR	18	19
29	H	19	20
30	CCP	20	21
31	MEI	3	3
32	CVP	21	22
33	IR	10	10

Fig. 16

Le tri effectué, la phase d'exécution peut débuter. Elle commence par l'initialisation que l'on choisit dans le menu

FIN	INITIA	ENRVAR	EXEC	RESUL

en pointant INITIA avec le réticule.

Cette initialisation n'est pas obligatoire, mais les résultats risquent d'être décevants si elle n'est pas effectuée car toutes les valeurs sont alors nulles.
L'initialisation est effectuée après que l'on ait choisi EXEC dans le menu :

FIN	LISTE	MODIF	EXEC

Les autres choix permettant d'obtenir la liste des valeurs initiales (figure 17) ou de modifier des valeurs.

```
BM      =     0.2778E 05
PDM     =     0.0000
MEI     =     0.0000
NM      =     0.0000
MLI     =     0.0000
IM      =     0.0000
DRIM    =     0.0000
MTS     =     0.0000
VP      =     0.2000E 05
IR      =     0.0000
NP      =      100.0
PLI     =      22.22
CP      =     0.0000
PLC     =     0.0000
SP      =     0.0000
RR      =     0.0000
IP      =     0.0000
DR      =     0.0000
H       =     0.2010E 05
CCP     =     0.0000
CVP     =     0.9950

POUR LA SUITE TAPER 'CR'
```

Fig. 17

L'initialisation effectuée, on peut choisir un ensemble de variables à enregistrer en désignant ENRVAR. Si aucun choix n'est fait toutes les variables sont enregistrées par défaut. L'exécution est alors lancée après que l'on est donné DT et DUREE.

Cette étape menée à bien, on peut avoir accès aux résultats en désignant RESUL dans le menu

FIN	INITIA	ENRVAR	EXEC	RESUL

E. Résultats

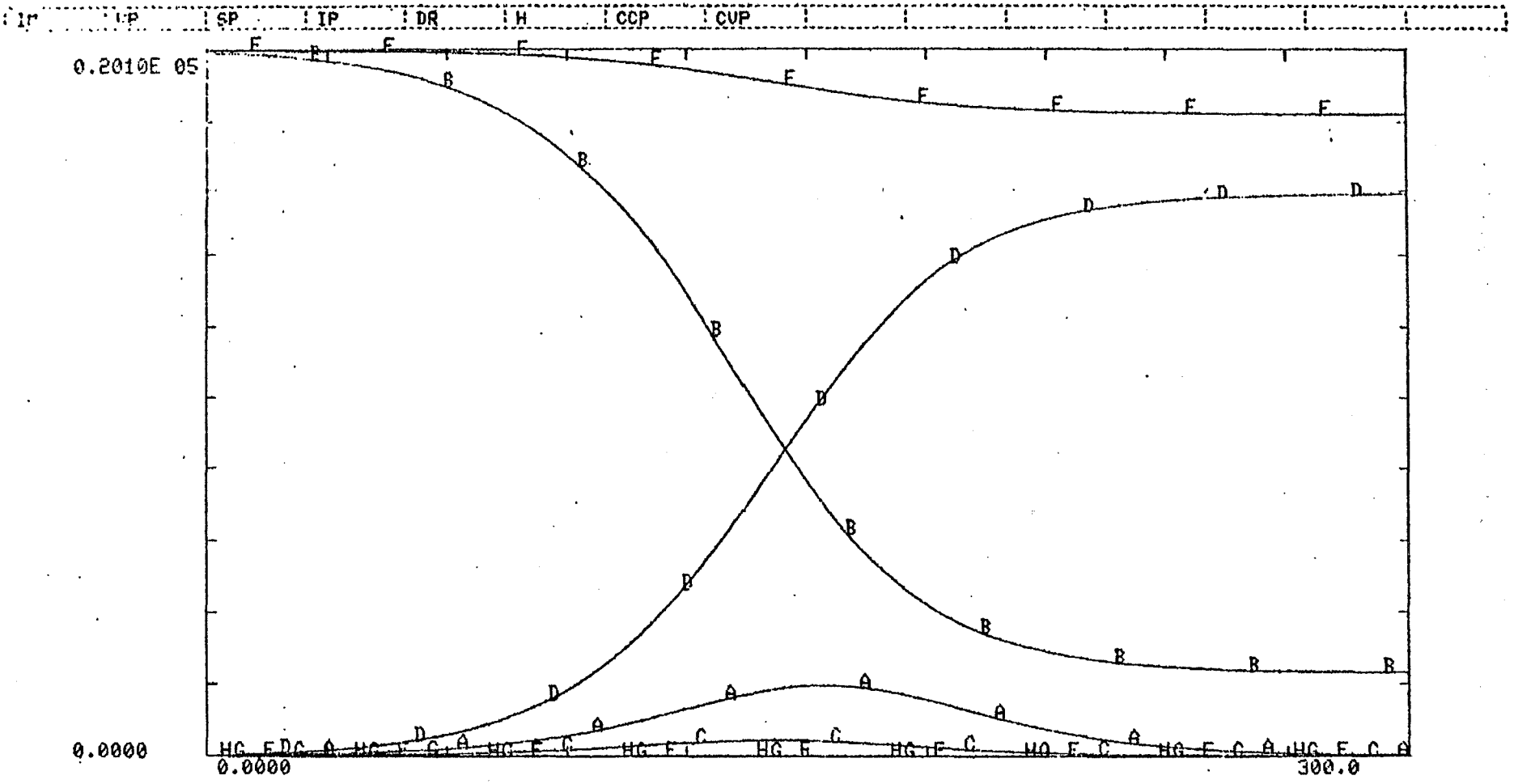
On peut choisir deux modes de présentation:

- courbes
- tableau de valeurs

Ce choix est permis par le menu

FIN	COURBE	VALEUR

Pour les courbes on peut choisir la mise en page interactivement et faire divers essais (figures 18 à 20)



A	IM	MIN	0.0000	MAX 1971.
B	VP	MIN	2335.	MAX 0.2000E 05
C	SP	MIN	0.0000	MAX 451.8
D	IP	MIN	0.0000	MAX 0.1598E 05
E	DR	MIN	0.0000	MAX 18.07
F	H	MIN	0.1832E 05	MAX 0.2010E 05
G	CCP	MIN	0.0000	MAX 0.4239E-01
H	CVP	MIN	0.1274	MAX 0.9950

Fig. 18

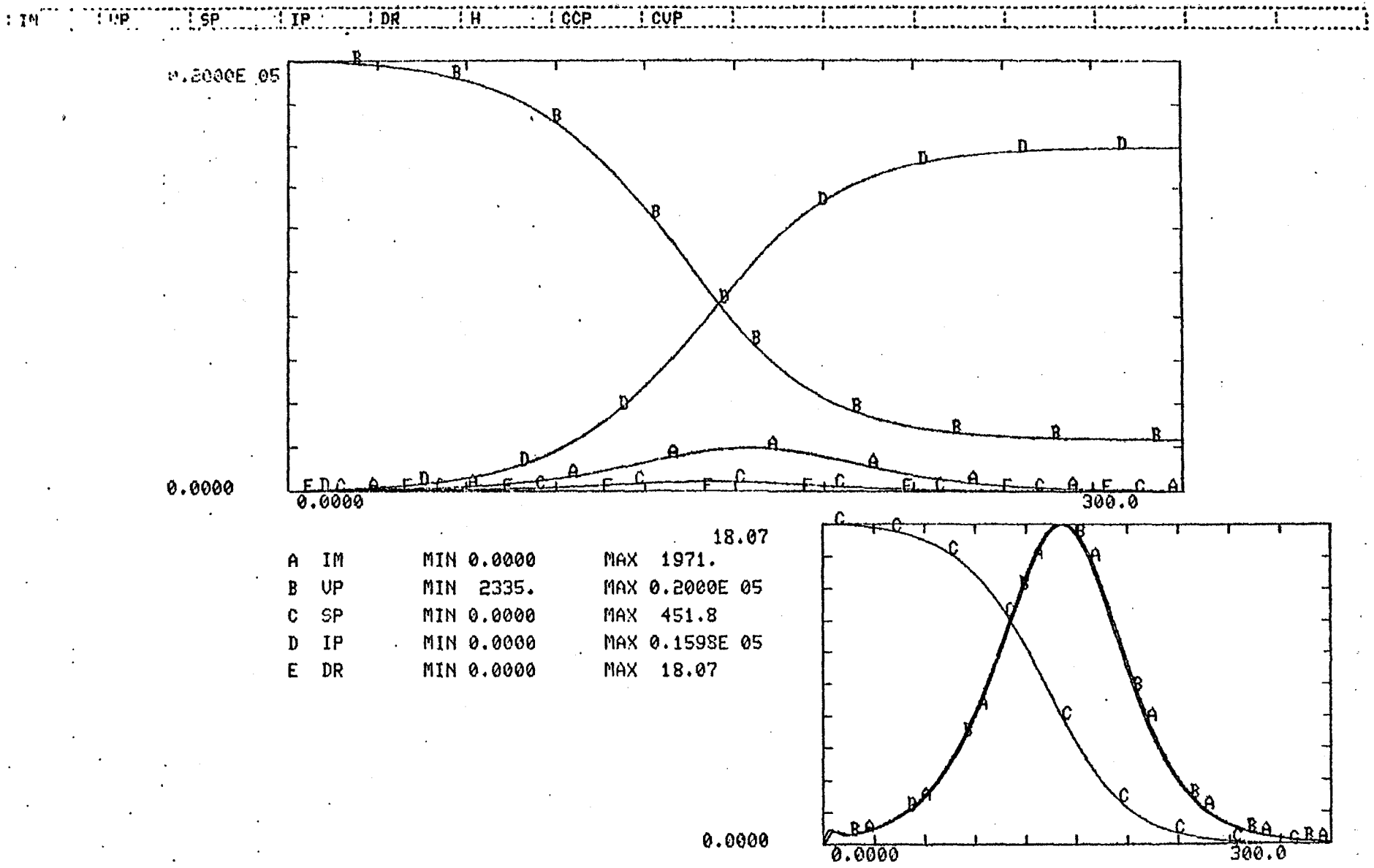

IM
VP
SP
IP
DR
H
CCP
CVP
0.2000E 05
0.0000
0.0000
300.0
18.07
A IM MIN 0.0000 MAX 1971.
B VP MIN 2335. MAX 0.2000E 05
C SP MIN 0.0000 MAX 451.8
D IP MIN 0.0000 MAX 0.1598E 05
E DR MIN 0.0000 MAX 18.07
0.0000
0.0000
300.0
A DR MIN 0.0000 MAX 18.07
B CCP MIN 0.0000 MAX 0.4239E-01
C CVP MIN 0.1274 MAX 0.9950

Fig. 19

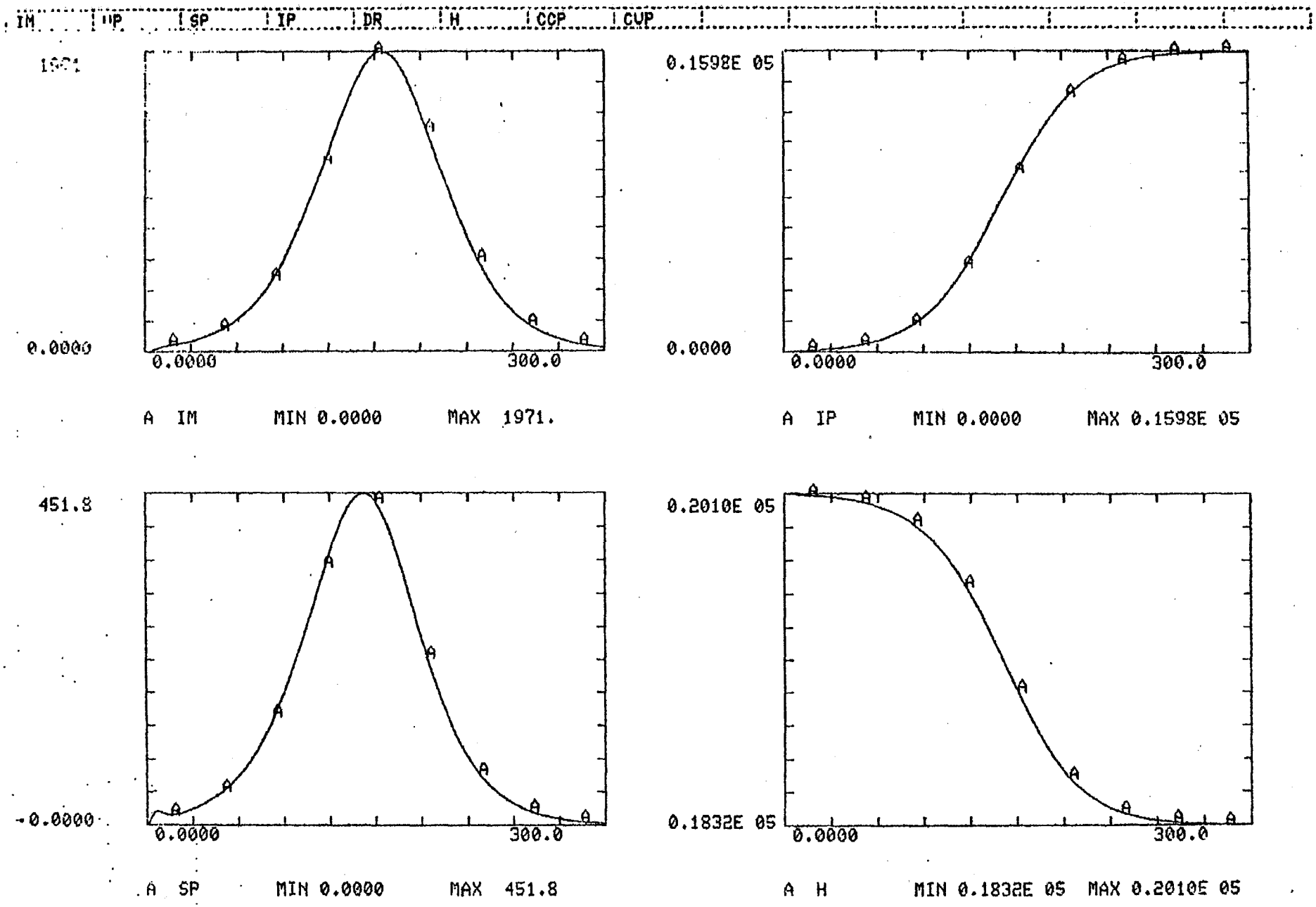

IM
IP
SP
IP
DR
H
CCP
CVP
1971
0.0000
0.0000
300.0
A IM MIN 0.0000 MAX 1971.
0.1598E 05
0.0000
0.0000
300.0
A IP MIN 0.0000 MAX 0.1598E 05
451.8
-0.0000
0.0000
300.0
A SP MIN 0.0000 MAX 451.8
0.2010E 05
0.1832E 05
0.0000
300.0
A H MIN 0.1832E 05 MAX 0.2010E 05

Fig. 20

La présentation sous forme de tableau est en figure 21. Remarquons qu'on peut choisir pour l'impression:

- l'instant de début d'impression
- la fin d'impression
- l'intervalle d'impression

(ces valeurs étant indiquées en nombre de DT et non pas en unité interne au modèle)

Dans l'exemple d'impression ces valeurs sont respectivement 0 6 1

```
**********************************************************************************

                        ****  MODELE  YELFEV  ****

**********************************************************************************

    IM              VP              BP              IP              DR
    H               CCP             CVP

**********************************************************************************

                      MINIMA DES VARIABLES

   0.0000           2335.          0.0000          0.0000          0.0000
   0.1832E 05       0.0000         0.1274

**********************************************************************************

                      MAXIMA DES VARIABLES

    1971.           0.2000E 05      451.8           0.1598E 05      18.07
   0.2010E 05       0.4239E-01      0.9950

**********************************************************************************

**********************************************************************************

                    ****  T =   0.0000      ****

   0.0000           0.2000E 05      0.0000          0.0000          0.0000
   0.2010E 05       0.0000          0.9950

**********************************************************************************

                    ****  T =   0.5000      ****

   0.0000           0.2000E 05      0.0000          0.0000          0.0000
   0.2010E 05       0.5528E-03      0.9950

**********************************************************************************
```

Fig. 21

```
   0.0000          0.2000E 05       1.235          0.0000          0.4938E-01
   0.2010E 05      0.9827E-03       0.9950

**********************************************************************************

                   ####  T =    1.500      ####

   0.3179E-01      0.2000E 05       3.182          0.2222          0.1273
   0.2010E 05      0.1310E-02       0.9950

**********************************************************************************

                   ####  T =    2.000      ####

   0.1616          0.2000E 05       5.472          0.7951          0.2189
   0.2010E 05      0.1553E-02       0.9950

**********************************************************************************

                   ####  T =    2.500      ####

   0.4556          0.2000E 05       7.846          1.780           0.3138
   0.2010E 05      0.1726E-02       0.9950

**********************************************************************************

                   ####  T =    3.000      ####

   0.9664          0.2000E 05       10.13          3.192           0.4052
   0.2010E 05      0.1841E-02       0.9950

**********************************************************************************
```

Fig. 21 suite

Voilà, notre modèle a été construit et exécuté. Les résultats sont enregistrés sur fichier. Nous avons dit précédemment qu'ils étaient accessibles à d'autres programmes pour des traitements divers au choix de l'utilisateur. Un exemple d'un tel traitement est donné en figure 22. C'est la surface de réponse d'une variable aux variations d'un paramètre. En l'occurence sur cette figure, nous avons en abscisse T, en ordonnée SP et en profondeur le paramètre NMPB variant de 0.5 à 1.5 avec un pas de 0.1

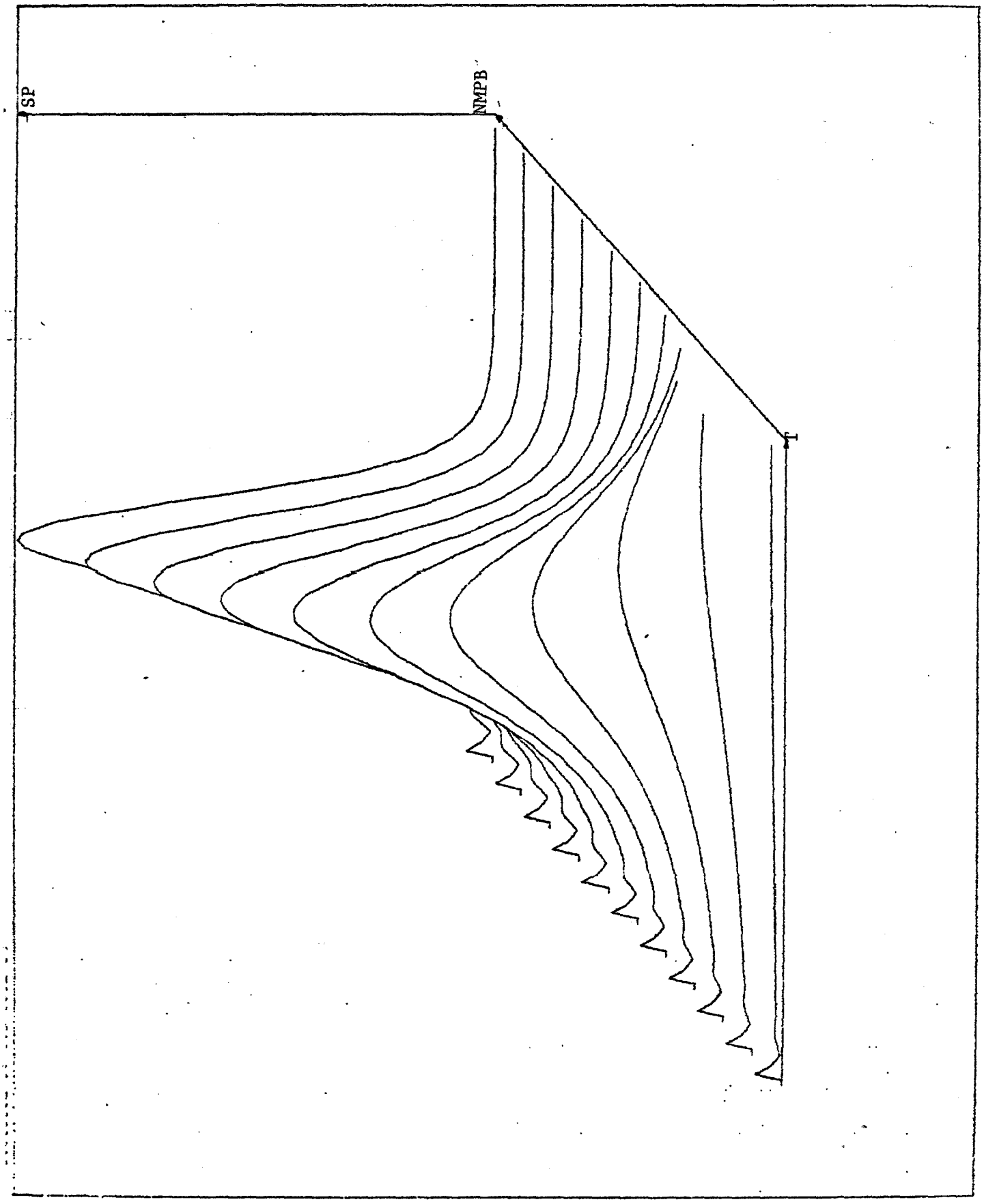


Fig. 22

Annexe: l'environnement matériel.

Le matériel sur lequel a été développé Dynamine graphique se compose:

-d'un mini ordinateur 16 bits PDP 11-40 (DEC)
comportant:
128 Koctets
2 drives de disques de 2.5 Moctets

-d'un terminal graphique TEKTRONIX 4015 (tube mémoire)

-d'un terminal servant au hard-copy: imprimante electrostatique GOULD.

Le logiciel tourne sur une partition de 28 kmots grâce à de l'overlay. L'image mémoire complète fait environ 80 kmots.

Ce logiciel a été écrit complétement en FORTRAN (DEC système RSX11M) sauf pour l'interface avec l'écran qui se compose de 20 lignes assembleur.

Bibliographie

Henri ATLAN "L"organisation biologique et la théorie de l'information"
Herman,Paris,72

BURNS "Converting signed digraphs to Forrester schematics and converting Forrester schematics to differential equations"
IEEE transactions on Systems,Man and Cybernetics,oct. 77,vol. SMC-7,n°10

"an algorithm for converting signed digraphs to Forrester schematics"
IEEE trans. on SMC,mars 79,vol. SMC-9,n°3

Noam CHOMSKY "reflexion sur le langage"
Maspero,Paris,77

P. FAURRE et M. DEPEYROT
"Eléments d'automatique"
Dunod,Paris,74

George S. FISHMAN
"Concepts and methods in discret event digital simulation"
Springer-Verlag,N.-Y.,77

Jay FORRESTER "Principles of systems"
Wright Allen Press,Cambridge USA,71
"Industrial dynamics"
MIT Press,61
"Urban dynamics"
MIT Press,70
"World dynamics"
MIT Press,

K. S. FU "Syntactic pattern recognition, application"
Springer-Verlag,N-Y,77

M. R. GOODMAN
"Study notes in system dynamics"
Wright Allen Press,74

GORDON "System simulation"
Prentice Hall,69

W. HUDITZ "An interactive graphic programming system for the construction of dynamic simulation models"
Concepts and tools of computer-assisted policy analysis,Birkhäuser,77

V. JARSCH "DYNASIS a graphic version of DYNAMO"
74

J. KLEIJNEN
"Statistical techniques in simulation"
Marcel Dekker,NY,74

LEDUC-LEBALLEUR,LUCAS,MARTINEZ
"Conception et réalisation d'un système graphique interactif indépendant du contexte d'utilisation: le logiciel de base GRIGRI"
RAIRO informatique,Vol. 12,n°2,78

LUCAS "Contribution à l'étude technique de communications graphiques avec un ordinateur. Eléments de base des logiciels graphiques interactifs"
Thèse doctorat d'état,Grenoble,dec. 77

J.W. MEERMAN
"interactive simulation langage THTSIM"
user's manual,dec. 78

E. MORIN "Le paradigme perdu: la nature humaine"
Seuil,Paris,73

T.H. NAYLOR "Computer simulation techniques"
J. Wiley and sons,NY,66

G.J. RITCHIE and J.A. TURNER
"Input devices for interactive graphics"
75

K.E. SAHIN
"Equivalence of markov models to a class of system dynamic models"
IEEE trans. on SMC,juil.79,Vol. SMC-9,n°7

B. WALLISER "Systèmes et modèles"
Seuil,Paris,77

J.N. WARFIELD
"Societal systems"
J. Willey and sons,NY,76

B. P. ZEIGLER
"Theory of modelling and simulation"
J. Willey and sons,NY,76

R. SHANNON "System simulation"
Prentice Hall,75

H. HAKEN "Synergetics"
Springer-Verlag,Berlin,78

J. BRESSY,J. MERMET,P. UVIETTA
"Langage de description de l'organisation et de la dynamique des systèmes socio-économiques et environnementaux (ODYSSEE)"
RAIRO informatique,Vol. 13,n°1,79

J. POPPER "La dynamique des systèmes"
Les éditions d'organisation Eyrolles,73